AFRL-RI-RS-TR-2008-110
Final Technical Report
April 2008

HUMAN AUGMENTATION OF REASONING THROUGH PATTERNING (HARP)

General Dynamics

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the Air Force Research Laboratory Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2008-110 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/ /s/

DANIEL E. DASKIEWICH
Work Unit Manager

JOSEPH CAMERA, Chief
Information & Intelligence Exploitation Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
APR 08	Final	Dec 02 – Dec 07

4. TITLE AND SUBTITLE

HUMAN AUGMENTATION OF REASONING THROUGH PATTERNING (HARP)

5a. CONTRACT NUMBER
F30602-03-C-0001

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
31011G

6. AUTHOR(S)

Stephen J. Sickels

5d. PROJECT NUMBER
GENO

5e. TASK NUMBER
A0

5f. WORK UNIT NUMBER
01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
General Dynamics
1400 Key Blvd, Ste 700
Arlington VA 22209-1546

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RIED
525 Brooks Rd
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2008-110

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# AFRL-08-0344

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The objective for this effort was to develop effective software tools to support collaborative intelligence analysis. The development led to two separate tools, tag/Connect and Catalyst. These tools can be used separately or combined for a more expansive collaborative capability. Tag/Connect allows analysts to apply "tags" (keywords) to Web-based resources, and to see and leverage the tags and tagged resources of others. Catalyst is a modeling tool that can be utilized for performing risk assessment and option generation. Catalyst is flexible and can be applied to a variety of intelligence issues. Catalyst models consist of nodes of information organized into hierarchical tree structures. Nodes can contain attachments or links to tags from tag/Connect. Tag/Connect is a core service on the three Intelink networks. .

15. SUBJECT TERMS
Collaborative analysis, Risk Modeling, Context grounded conversations

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Daniel e. Daskiewich
U	U	U	UU	52	19b. TELEPHONE NUMBER (*Include area code*) N/A

Table of Contents

List of Figures

Executive Summary

This report provides an overview of the work accomplished by the General Dynamics Advanced Information Systems Team on AFRL Contract F30602-03-C-0001. The high-level goal for our effort was to develop effective computer-based support for collaborative intelligence analysis.

We describe three collaborative software tools we have developed:

- *tag|Connect* allows analysts to apply "tags" (keywords) to Web-based resources, and to see and leverage the tags and tagged resources of others. tag|Connect is a Core Service on the three Intelink networks, and is growing steadily in use.
- *Catalyst* is a flexible application for individually and collaboratively representing a wide range of intelligence issues, problems, and challenges. Catalyst "models" consist of nodes of information organized into hierarchical tree structures. Nodes can contain "attachments" (links to Web-based resources) and thus are ideal for organizing resources—as well as for developing an analysis supported by available resources. Catalyst supports both "individual" and "collective" models, as well as the sharing of information among models.
- *Context Grounded Conversations (CGCs)* are computer-supported communications that are explicitly linked to specific content represented within an application. A key aspect of CGCs is that they extend "conversation" mechanisms that are already in use, rather than providing an embedded communications mechanism that functions only within the context of a particular application.

For each tool we describe the goals and objectives addressed. We describe also the processes and insights that guided each tool's development, and lessons learned.

We describe our "Develop / Understand / Improve" cycle, and its central importance in our development process. Through this approach we seek not only an understanding of the technical issues associated with tool development, but also of the collaborative and analytic issues as well. We apply the understanding gained directly back in to our overall development process. Toward that end, we have consistently paired technologists and domain experts in experimentation, which we believe is a truly transformative approach to developing effective computer-based support for intelligence analysis.

In particular, we have conducted a number of experiments over the life of our effort in which graduate students and researchers from the Monterey Institute for International Studies have put our tools to use in addressing specific intelligence challenges. We describe these experiments and the insights we have gained from them.

Finally, we describe three key unifying themes that have emerged from our work and that drive our work:

- *Blending the Personal and the Collective*: Encouraging and supporting collective benefits from individual work.
- *Capturing Diversity*: Assisting analysts in capturing constructive conflict.
- *Context-Grounded Consensus Formation*: Providing tools to support analysts in grounding analytic negotiations, requests, and discussions in the context of shared explicit analytic artifacts.

1 Overview

This report provides an overview of the work accomplished by the General Dynamics Advanced Information Systems Team on AFRL Contract F30602-03-C-0001. The high-level goal for our effort was to develop effective computer-based support for collaborative intelligence analysis.

This contract ends on 17 December, 2007. However, the General Dynamics Team will continue to leverage and build upon key technologies developed on this contract and described herein. Thus, this report both describes our accomplishments to date and sets the stage for future work.

Our overall research and development approach on this effort can be characterized as a "Develop / Understand / Improve" cycle, the "Understand" phase of which is the true foundation of our work. To ensure that the understanding we gained was both deep and well-integrated, we have *paired technologists and domain experts in experimentation*, which we believe is a truly transformative approach to developing effective computer-based support for intelligence analysis.

Throughout our effort, domain experts, including Gary Ackerman of the Monterey Institute for International Studies (MIIS)—and, currently, the Center for Terrorism and Intelligence Studies (CETIS)—and Steven Simon, formerly with Cyberneutics, Inc., have participated directly as integrated members of our team. Mr. Ackerman is a recognized expert on terrorism studies, and Mr. Simon is a former senior member of the National Security Council.

We have also conducted a number of experiments in which proxy Intelligence Analysts—such as researchers and graduate students with MIIS—have used prototypes of our software tools over periods ranging from several days to several months to collaboratively address specific analytic taskings. We believe that the feedback and insights we have received via these experiments has been central to our success.

The "Understand" phase of our development cycle has focused on the following three dimensions of computer-based support to intelligence analysis—all of which are of critical importance:

> Collaborative Foundations: This dimension includes issues such as consensus, diversity, avoiding "groupthink," negotiation, and workflow.
>
> Analytic Foundations: This dimension includes various aspects of the intelligence analytical cycle, such as collection, collation, evaluation, and reasoning.
>
> Technical Foundations: This dimension includes issues such as our tools' usability and utility, flexibility, and generalizability—as well as interoperability and support for Web Services-based and SOA-based integrations and "mashups."

Although we have developed a substantial body of *technical* accomplishments, we would like to highlight a set of *conceptual foundations* that we developed over the life of this contract which we believe have been instrumental in guiding and informing our technical

accomplishments. We developed these conceptual foundations through our focus on *understanding* collaborative analytic dynamics. These conceptual foundations, or themes, are summarized as follows:

Blending the Personal and the Collective: We have designed our tools so that an analyst's individual work can "automatically" benefit others. In general, this directly addresses the "zero sum" problem inherent in most collaborative approaches, in which individuals tend to perceive collaborative work as separate and distinct from their "normal" workflow, and therefore tend to avoid it.

Capturing Diversity: General Dynamics' tools developed on this effort are designed to capture "collective diversity," by which we mean the disagreement, dissent, and constructive conflict that we believe are necessary drivers of effective collaboration. We believe that the Intelligence Community can practice effective computer-enabled collaboration only if the tools it uses support this requirement directly.

Context-Grounded Consensus Formation: Capturing diversity is of critical importance for the Intelligence Community—but so also is developing a meaningful consensus from that diversity. The General Dynamics Team has developed an approach to "Context-Grounded Conversations" (CGCs) that supports analysts in working together to develop a meaningful consensus. Our approach allows analysts to *ground* analytic negotiations, requests, and discussions (collectively, "conversations") in the context of shared, explicit analytic artifacts (such as Catalyst models and items tagged in tag|Connect, both of which are described below). These contextually-grounded conversations allow a group to align its views and conclusions, while simultaneously capturing—in the analytic artifacts—conflicting views and constructive conflict within a shared and coherent analytic framework. By allowing analysts to interlink analytic artifacts and "conversations" about the artifacts, we directly support a productive tension—and a productive balance—between consensus and diversity.

Because these conceptual themes were developed within the context of our overall "Develop / Understand / Improve" *cycle*, they both drive and are driven by our technology developments and the experimentation we undertake with our technology.

2 Technologies

The General Dynamics Team has developed and implemented a number of software tools over the life of this effort, including tag|Connect, Catalyst, and Context-Grounded Conversations. We begin with a description of tag|Connect:

2.1 tag|Connect

2.1.1 Overview

tag|Connect, shown in *Figure 1*, is a flexible and easy-to-use Web application that allows analysts and others to organize Web-based resources using "tags" (or keywords) of their own choosing. Analysts can use tag|Connect to quickly assign multiple tags to resources and can then use an intuitive interface to browse or search those tags and tagged resources.

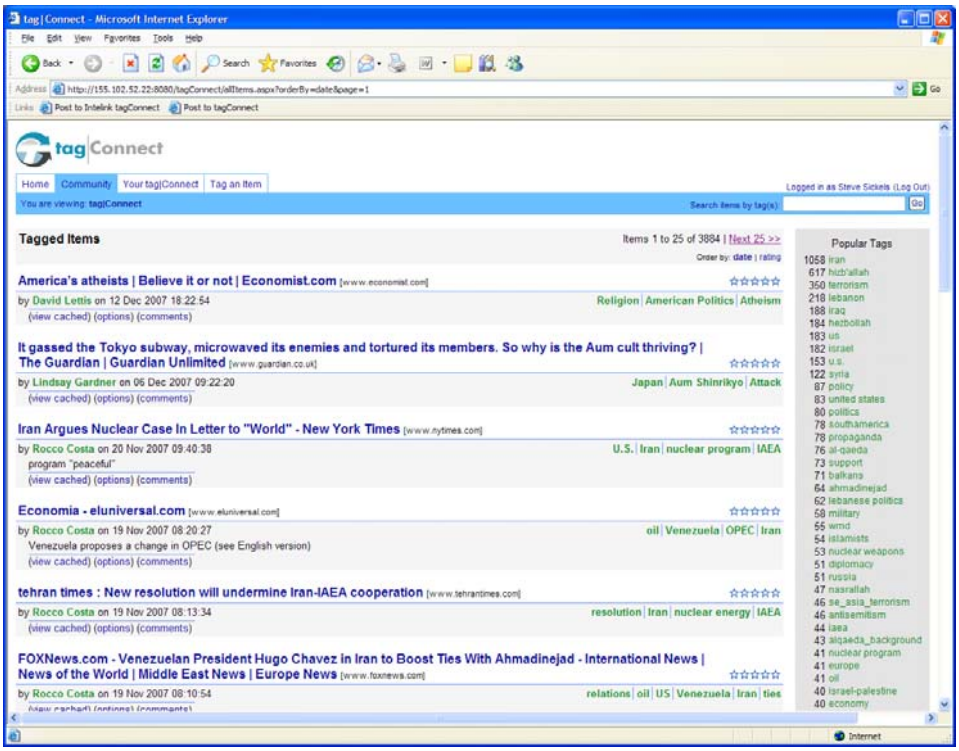

Figure 1. tag|Connect

Because tags are *shared* (unlike traditional browser bookmarks), tag|Connect also provides tremendous collective benefits by giving analysts the opportunity to leverage the work of others. Analysts can view other analysts' tag|Connect tags and tagged resources to find additional key resources relevant to their own work. They can quickly understand topics and analytic subjects of broad or specific interest among peers and colleagues in the Intelligence Community.

It is easy within tag|Connect to switch to a view that shows only a filtered set of tags and tagged resources. For example, an analyst can choose to view only the tags she has used and the resources to which she has applied them. Additionally, she can view resources tagged with a particular tag (or tags).

Users can optionally add comments to tagged resources, which is especially useful when sharing with others. Users can also rate tagged resources using a familiar five-star rating system (as on Amazon, Netflix, etc.), and can supply optional comments or justifications in association with their ratings.

tag|Connect is a powerful tool for organizing resources and for discovering the resources and topics of interest to and in use by others. As such, it allows analysts to:

- Organize resources according to individual preferences, which optimizes recall and retrieval; and to
- Leverage the collective organizational space, which provides a diversity and breadth and relevant resources.

tag|Connect is currently deployed as a "Core Service" on the three Intelink networks: JWICS, SIPRNet, and Intelink-U. *Figure 2*, below, shows tag|Connect on Intelink-U.

tag|Connect is in the process of being deployed to the Open Source Works (OSW), and it is a planned component of the ODNI's forthcoming "A-Space" analytic environment. On the three Intelink networks it is currently utilized by thousands of users, with its popularity growing rapidly.

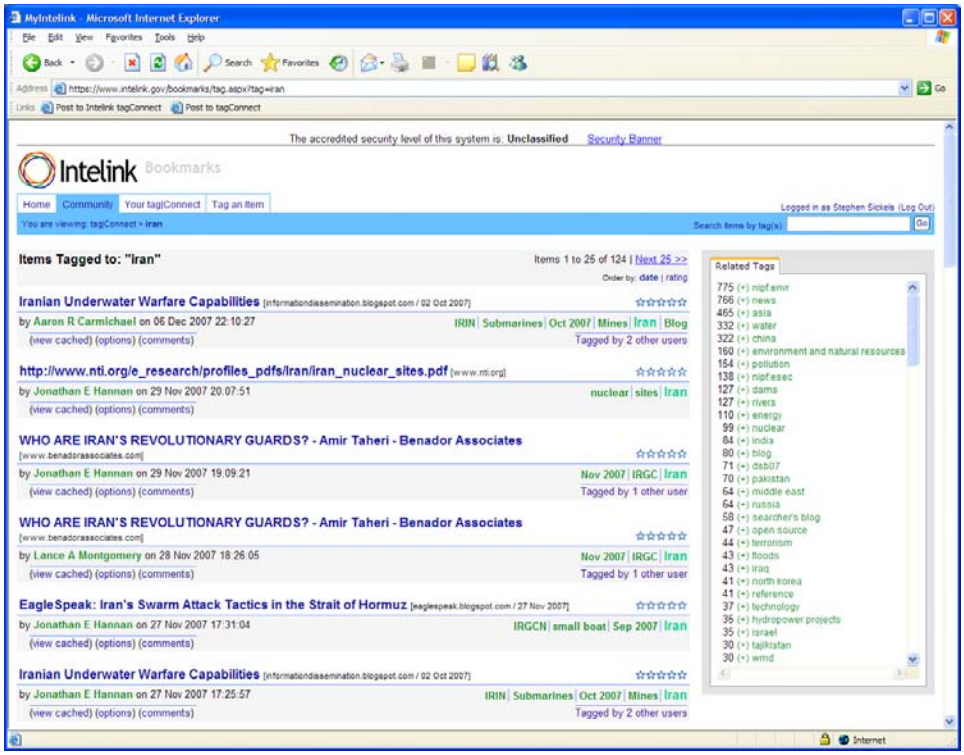

Figure 2. tag|Connect on Intelink-U

2.1.2 Deployment Support

The General Dynamics Team has worked closely with the Office of the Director of National Intelligence's (ODNIs) Intelligence Community Enterprise Services (ICES) office, which is responsible for running and maintaining the Intelink networks, to ensure that tag|Connect meets their requirements and the requirements of the Intelink user community.

During our initial coordination meetings with ICES[1] on 6 June and 26 July, 2006, we discussed two options for managing tag|Connect's deployment and development life cycle: the first option was that the General Dynamics Team would provide ICES with the tag|Connect source code, and ICES would then be responsible for any "hardening" necessary to support a large-scale (world-wide) deployment as well as any modifications or bug fixes that might be necessary following the initial deployment. The second option

[1] At that time ICES operated as the Intelink Management Office, or IMO.

was that the General Dynamics Team would "harden" tag|Connect for a world-wide deployment and would be responsible for any modifications that became necessary or for implementing requested feature and functionality additions. We mutually agreed that the second option was best, for several reasons:

- As the General Dynamics Team planned to continue improving tag|Connect through our research, this arrangement ensured that we could easily fold enhancements and new features into the ICES code baseline, thus providing the benefits of our continued research back to the Intelink user community.
- The "partnering" of developers and users is critical to effective research! As maintainers of the tag|Connect code baseline, the General Dynamics Team was then responsible for addressing the community's feature and enhancement requests, which assisted us greatly in guiding our research and development agenda.

Several of the modifications we undertook, guided by and in consultation with ICES, were:

- Developing an interface to ICES's "Passport" log-in and authentication service;
- Transitioning to a GUID (Globally Unique IDentifier) referencing approach in our database;
- Implementing significant database optimizations to ensure that tag|Connect performed well in a high user-load environment; and
- Making modifications to tag|Connect's Data Access Layer (DAL) to ensure that it would function properly in ICES's "server farm" environment, in which users may be routed to different physical machines in the "farm" during a single session.[2]

On 2 October 2006 we delivered tag|Connect v0.5.0 to ICES. On 30 October, ICES opened up tag|Connect for a limited beta test for small set of Intelink users. Then, on 5 February, 2007, tag|Connect went "live" as a Core Service, available to all Intelink users on all three networks, worldwide.

We have also supported two additional deployments of tag|Connect, each of which has necessitated some tailored modifications:

- Research and Development Experimental Collaboration Program (RDEC) testbed: For the RDEC, we made some modifications to account for several Active Directory Issues specific to the RDEC's LDAP servers (for example, some RDEC users don't have email addresses, which tag|Connect was expecting). Our first delivery to the RDEC was v0.8.0 on 2 January, 2007.

[2] Generally, Web application developers can assume that a single server will manage a single user's "session." This allows the developer to assume that session information ("state") can be stored on the server, which simplifies the development process. In a server farm environment with round-robin load balancing (such as is used on Intelink), however, such state information cannot be stored and accessed.

- Open Source Works (OSW): For our OSW delivery, a number of modifications were necessary to support their login/authentication approach, as well as to introduce new features as requested, such as RSS 2.0 feeds and Unicode character set support. Our first delivery to OSW was v0.9.0 on 28 September, 2007.

2.1.3 Continued Development and Improvements

The most significant improvement we have made to tag|Connect was a near-complete reimplementation of the User Interface (UI) in Asynchronous JavaScript and XML, or AJAX. Version 0.9.0 was the first AJAX-based release, which we provided to ICES on 9 August, 2007. AJAX offers great potential for increasing the functionality, responsiveness, and interactivity of Web applications.

Details of our challenges and accomplishments in moving tag|Connect to AJAX are presented in the paper *tag|Connect: Extending the Breadth of Social Software on Intelink*, by John Nolley II, which was developed in association with Mr. Nolley's presentation at the 2007 Intelink Technical Exchange (ITE). The ITE was held in Falls Church, Virginia, on 13 June, 2007.

Our AJAX-based tag|Connect release included a significant new feature: a five-star ratings capability, shown in *Figure 3*, which is quite similar to the five-star ratings on Amazon, Netflix, and numerous other websites. The average rating to date (if any) is shown for each tagged resource, and users may easily record or modify their ratings by

Figure 3. tag|Connect's rating and comment feature

clicking on the desired number of stars in the rating display. Users can also add (or edit) an optional comment/justification in association with their rating.

It is easy for users to see the ratings applied by others: when a user hovers her mouse over a previously-rated item, she is presented with a pop-up window that allows her to

drill in to see who rated the item previously, what the ratings were, and any associated comments/justifications.

Furthermore, rating an resource and tagging an resource are completely separate functions: anyone logged in to tag|Connect can rate *any* resource they see, regardless of who tagged it.

2.1.4 Web Services

We have architected the tag|Connect application to have a clean and well-defined separation between its underlying "engine" (or "business logic") and the Web-based user interface (UI) that exposes its functionality to users. Although the tag|Connect UI supports users in easily navigating and exploring the underlying "tag space," there is no reason why tags need to be exposed *only* through the tag|Connect UI. There is, in particular, great potential advantage in tags—and ratings—being exposed to and by *other* application UIs.

So, for example, the developers of any application that presents URL-based content to users could create additional UI functionality that would allow users to tag and/or rate that content from *within that application*. Or perhaps to see what tags or ratings others might have applied previously. Although the application's developers would have to implement new UI functionality to support this capability, there would be no need for the developers to re-implement a tagging "engine" to manage the tag data. Instead, tag|Connect's underlying engine (which provides the capabilities to create and access tag data) could easily be accessed by the new application via calls to tag|Connect's Web Services API. (These calls are hidden from the users, of course.) An additional benefit in this approach—over and above saving the application's developers from having to re-implement a tagging engine—is that any tags applied to resources from within the new application's context are also "automatically" available from within the tag|Connect UI. And, conversely, resources listed or depicted in the new application can show tags previously applied from within tag|Connect as well.

To facilitate exposing tag data within other application contexts, we have developed an extensive Web Services API for tag|Connect. There are three different "levels" of calls that can be made against the API:

1. "Full" calls which are the most complex to construct, but are also the most powerful and flexible. These calls correspond roughly to those made programmatically from within tag|Connect against its underlying "engine."
2. "Simple" calls which hide some of the complexity of the full calls at the cost of some flexibility and functionality.
3. "REST" calls which are the simplest of all—but which are the least flexible. Where a "standard" query can be identified that can be called for multiple reasons without any modifications (and thus, flexibility is not needed), these are ideal. Due to "popular demand" from the tag|Connect user community, we have developed a set of REST calls, and will continue to add more as we can.

To encourage and facilitate the use of tag|Connect's Web Services API, the General Dynamics Team hosted a workshop, the "tag|Connect Development Mashup Workshop," on October 25, 2007, at its offices in Crystal City, Virginia, at which we provided an overview of the tag|Connect Web Services API. Speakers at the workshop included John Nolley of General Dynamics (tag|Connect's lead developer), as well as personnel from the ODNI's Intelligence Community Enterprise Services (ICES) office, who described work and ideas regarding how tag|Connect is and can be "mashed up" (via Web Services) with other offerings and applications on the Intelink networks, such as Intellipedia. There were approximately 40 attendees (both in person and via WebEx) representing a full spectrum of Intelligence Community agencies and organizations (including the ODNI, CIA, DIA, NSA, Army, and Marine Corps).

tag|Connect's Web Services API has also been put to good use by a number of contractors on IARPA's CASE (Collaboration and Analyst System Effectiveness) Program. These connections emphasize the fact that tag|Connect's Web Services API can expose tag|Connect's underlying data and functionality not only to different *user* application contexts, but also for consumption by applications that use the tag|Connect data to compute various results, rather than necessarily exposing the tag data directly to users.

For example, we developed a connection with the "CAST" application developed by Lockheed Martin, which develops internal models of users' evolving "contexts." The CAST application accesses the tag|Connect data via Web Services calls as well as receiving published "events" describing real-time changes to the tag|Connect data, and it provides back information based on its calculations—such as recommended tags for a resource—that were then displayed in a slightly-modified tag|Connect UI.

2.1.5 A Brief History of tag|Connect

The General Dynamics Team had long been observing the success of the del.icio.us (and other) tagging application on the open Internet, and became interested in trying to apply and adapt tagging to the Intelligence Community's needs in early 2005. In June of 2005, we proposed this idea to our government sponsors on this effort, and received authorization to proceed.

In August of 2005 we had our first prototype ready for use in the 2005 Experiment, as described in Section 3.5, below. Following the experiment we chose the name tag|Connect (and we registered the www.tagConnect.com domain name). *Figure 4* shows the version of tag|Connect used during the 2005 Experiment (which has been much improved!), as well as some of the items tagged during that experiment.

We continued developing tag|Connect, and in June of 2006 met with ICES, as described in Section 2.1.2, to discuss a possible deployment of tag|Connect on the Intelink Networks.

Details of tag|Connect's development following our June, 2006, meting with ICES are presented in Sections 2.1.2 and 2.1.3.

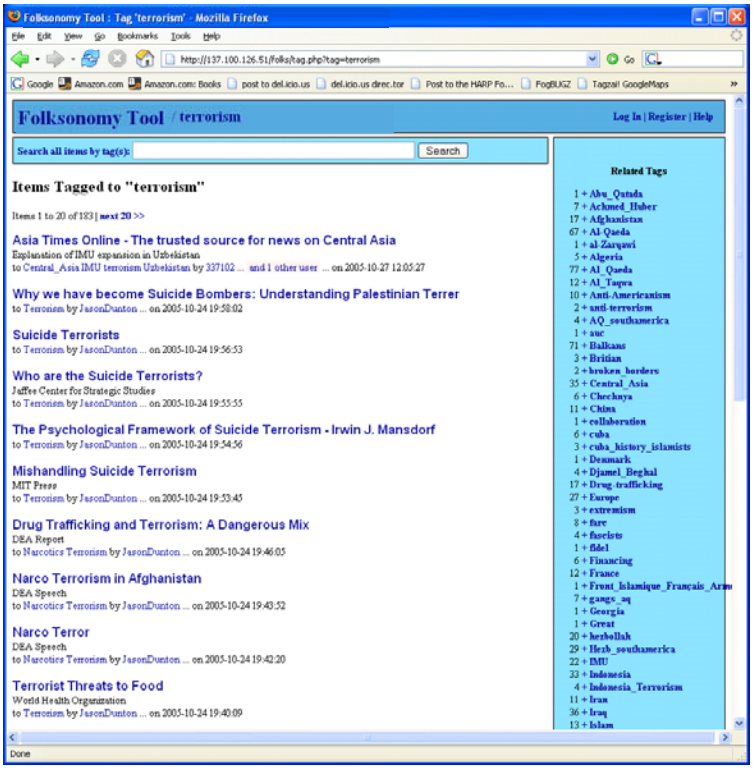

Figure 4. tag|Connect - an early version from August, 2005

2.2 Catalyst

2.2.1 Overview

General Dynamics' Catalyst, shown in *Figure* 5, is a flexible application for individually and collaboratively representing and analyzing a wide range of intelligence issues, problems, and challenges. Catalyst supports a variety of analytic activities: it is an effective tool for structuring information about a topic, and it is also an effective tool for conducting an analysis, evaluation, or assessment.

A Catalyst *model* consists of "nodes" containing text that are organized into hierarchical tree structures. Catalyst's "trees" support natural representations of a problem into sub-problems, or a system into subsystems, or an issue into sub-issues. Catalyst also supports "attachments": the URL of any URL-accessible information can be easily "dragged-and-dropped" from a Web browser into a node, automatically creating a hyperlink within the node to the URL-based information resource and thus allowing relevant URL-accessible information to be organized directly within the context of the issue, problem, challenge, or system being considered.

In addition to providing a flexible organizational and analytic framework for individual analysts at work, Catalyst also seamlessly extends analysis into the *collective* realm via a number of powerful collaborative features.

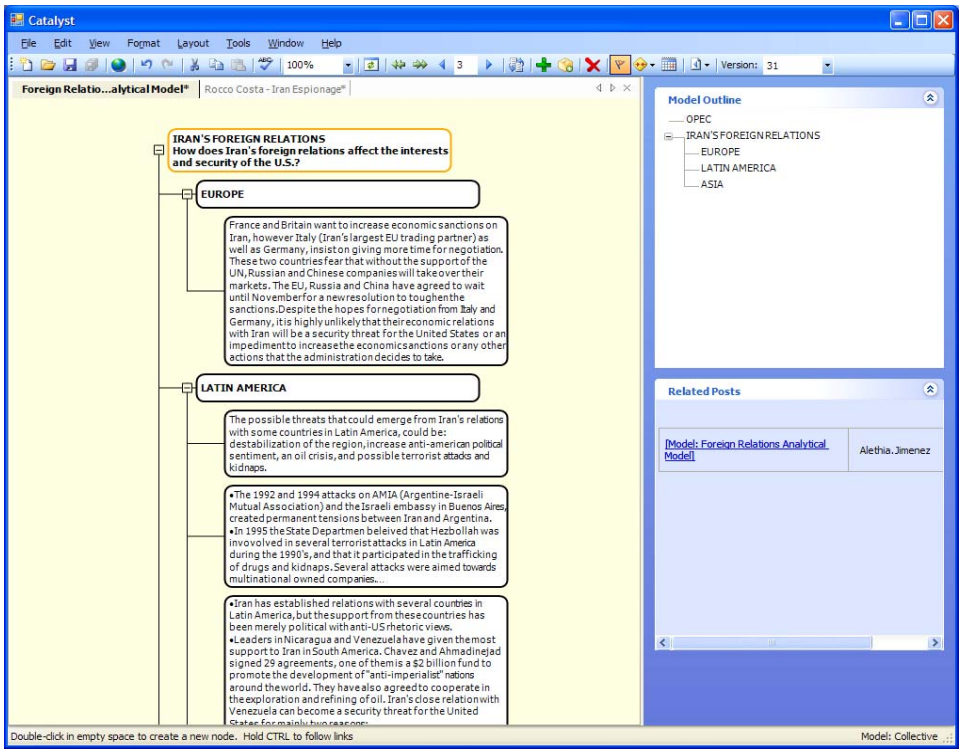

Figure 5. Catalyst

As a framework for collaborative work, Catalyst is a client-server application that stores multiple Catalyst models created by multiple individuals on a single server. Any individual with the Catalyst client on his or her computers and with log-in access to the server is able to open and view anyone else's Catalyst model.[3]

Not only can analysts view the informational and analytic structures created by others in Catalyst, they can also "adopt" (copy and paste) each other's work—that is, node structures and associated attachments—directly into their own models. Catalyst thereby enables analysts to *augment* and *extend* their own work with relevant portions of the work of others.

Catalyst provides two types of models:

- *Individual models* are created and fully controlled by a single individual.
- *Collective models* can be created, edited, or augmented by anyone who chooses to.

Thus, not only does Catalyst provide a common context for a range of analytic activities, but also provides a common context for both individual and collective work, thereby affording opportunities for interconnections and synergies among analysts' work. Furthermore, Catalyst allows analysts to easily shift back and forth between individual and collective tasks. (Most collaborative tools, by contrast, enforce a counterproductive

[3] Catalyst models are accessed via the familiar "File / New" and "File / Open" menu commands. However, they aren't actually files; they are data structures stored in a database. However, they *appear* as files to Catalyst's users and are likewise organized into "directories" by the same file metaphor.

"boundary" between collaborative and non-collaborative tasks and workflows, thus cutting off opportunities for collaborative synergies.) We believe that this "blending of the personal and the collective" can spur collaboration and collective benefits, even when such benefits are not explicitly intended.

In summary:

- Catalyst provides a framework that supports the analytic decision-making process.
- Catalyst enables analysts to develop problem- or task-specific organizational structures—"outlines"—that can be used for organizing existing, incoming, or discovered data and information. These structures can provide value not only to their creators, but also to others, as well (including via Catalyst's support for "adopting" content from one model to another).
- Catalyst supports a variety of workflows: analysts can develop and refine their structures in response to data and information that becomes available through an ongoing analysis, or they can develop their structures in advance to document a set of plausible alternatives and to provide a structure for organizing and assimilating incoming or discovered information relevant to those alternatives.

2.2.2 Features and Functions

Analysts can use Catalyst to organize information into "nodes" of text that are joined via parent-child relationships (links) into tree structures. A "root node" at the top of the tree generally represents the overall question, hypothesis, or subject being addressed.

The text in a Catalyst node can be short or long, ranging from catch-phrases to multiple paragraphs of text quoting from source materials or laying out the details of an analytic argument.

A single Catalyst "model" (that is, the units of data and information that are accessible via Catalyst's "File / Open" and "File / New" commands) can contain any number of trees, which can be arranged by users on the Catalyst "canvas." New nodes can easily be added as children of existing nodes (by a number of mechanisms, the simplest of which is probably to right click on a node and to select the "Add child" option). Also, new freestanding nodes can easily be added anywhere on the Catalyst canvas just by double-clicking on the canvas where the new node is to be positioned, and can then be dragged into existing trees or even serve as the foundation for "new" trees, or simply be left to stand alone. Nodes or trees can easily be moved around on the canvas by clicking and dragging.

As noted above, Catalyst models also come in two basic types:

- *Individual models*: These models can be modified only by the model's creator—although they are viewable by all.
- *Collective models*: These models can be modified by anyone with access to Catalyst.

Creators of individual models can, at any time, change any of their individual models to collective models, such that others can then participate together in their development.

In keeping with our general approach throughout our effort, we do not attempt to assume or prescribe how and when individual and collective models "should" be used within an analytic process. Instead, we believe that these choices are best made by Catalyst users or groups of users. (We do believe, however, that our *understanding* of the factors that drive these decisions is of key importance in our development efforts.) These choices regarding the use of individual and collective models may be made on a case-by-case basis, or they may develop over time into established "norms." For example, a given group or ad-hoc team might well decide to do all work on a particular issue in a collective model to which everyone can contribute, rather than making use of individual models for individual work. Nonetheless, the General Dynamics Team did want to ensure that if and when appropriate or desired, individuals could create models that others *cannot* change or alter, as well as to ensure that these individual models can provide the maximal collective benefits possible.

Catalyst makes extensive use of a "drag-and-drop" mode of interaction for nodes and trees:

- The trees and nodes can be easily dragged and repositioned on the canvas.
- Subtrees and nodes can be "detached" from their parent nodes and dragged to new locations, or they can be "attached" to any node, which then becomes their new parent.
- The nodes *within* a tree can be rearranged and reordered.
- "Attachments" (hyperlinks to Web resources) can be added to a node by drag-and-drop from any Web browser's address bar (by grabbing the small icon to the left of the URL in the browser's address bar). A node's attachments show up below the node's text as clickable hyperlinks.

(Any operation that can be accomplished via drag-and-drop can also be accomplished by copy-and-paste.)

Catalyst has another particularly powerful drag-and-drop feature: Text from a Web page can be highlighted and dragged onto the Catalyst palette, which then automatically creates a new node with that text in quotes. And, Catalyst captures the source URL and its title, which is automatically added as an attachment to the new node. This is shown in *Figure 6*: A portion of text from a Web page is shown in the upper left, and the Catalyst node that results when the text is highlighted and dragged into the Catalyst canvas is shown in the lower right. Note the blue hyperlink below the quoted text in the Catalyst node. This automatically-created hyperlink points back the source Web page from which the text was dragged.

This feature is designed to make the mechanics of Web-based research (including on Intranets such as Intelink) particularly fast and easy. Relevant information can quickly be added to a Catalyst model, with the source of that information automatically captured and available for easy access at any time in the future.

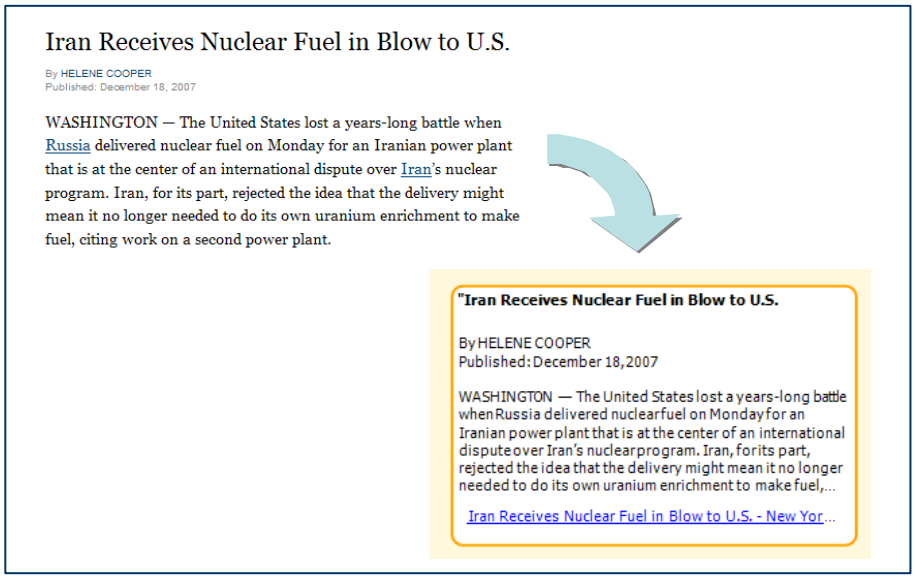

Figure 6. Catalyst's drag and drop of Web text

And finally, a key Catalyst feature is that individuals can "adopt" (via copy and paste) nodes from one model to any other model to which they have write access (that is, to any collective model or to any of their own individual models). If the copied node is pasted "onto" a node in the target model, it becomes a child of that node. If, on the other hand, it is pasted elsewhere on the canvas, it is just copied as a "free node" to that new location.

As with all drag-and-drop and cut-and-paste operations in Catalyst, if the copied node has children nodes, those children are automatically copies as well.

Catalyst provides a number of other features:

- Layouts: Catalyst trees can be viewed in three different layouts: indented, horizontal, and vertical. These layouts are easily selectable from both a drop-down menu and a toolbar icon, allowing analysts to easily select their preferred view—or to switch among views if desired. (Some analysts, for example, consistently prefer the vertical, or "org chart" view.) *Figure 7* and *Figure 8* show the indented and vertical layouts, respectively.
- Collapse/Expand: Catalyst includes a number of different mechanisms for controlling how much of and what portions of a model are in view at any time. There are collapse/expand points on each node that has children, identical to those used in Microsoft's Windows Explorer. There are also controls on the toolbar that allow entire trees to be expanded or collapsed one level at a time.
- Attribution: Catalyst captures attribution for all nodes. If someone adopts material from someone else's model, then that material is forever attributed to its creator. Likewise, as people work together to construct collective models, attribution of each node is always captured.

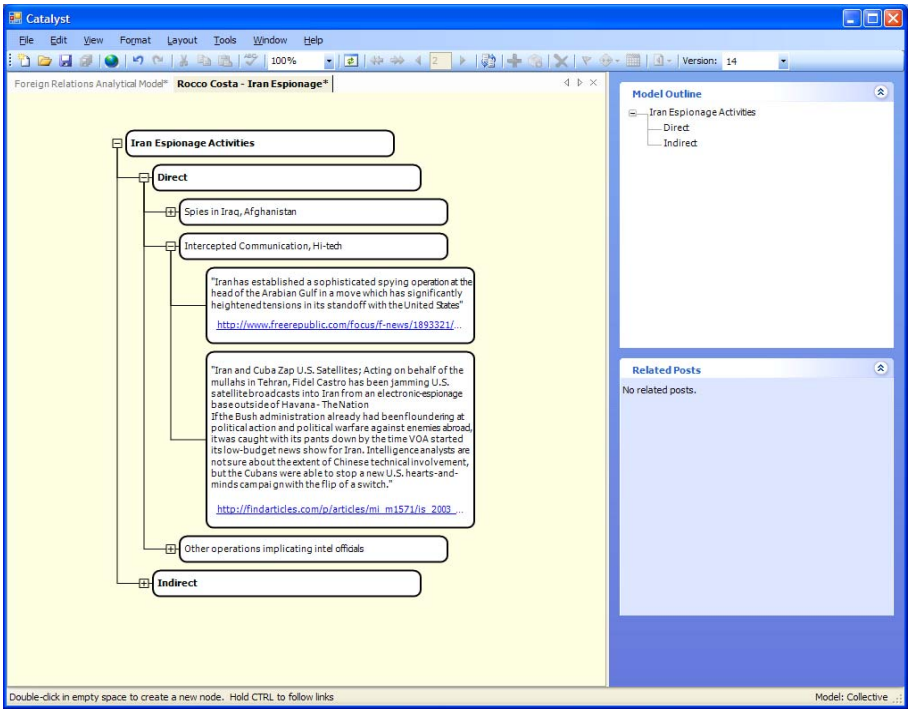

Figure 7. Catalyst - indented layout view

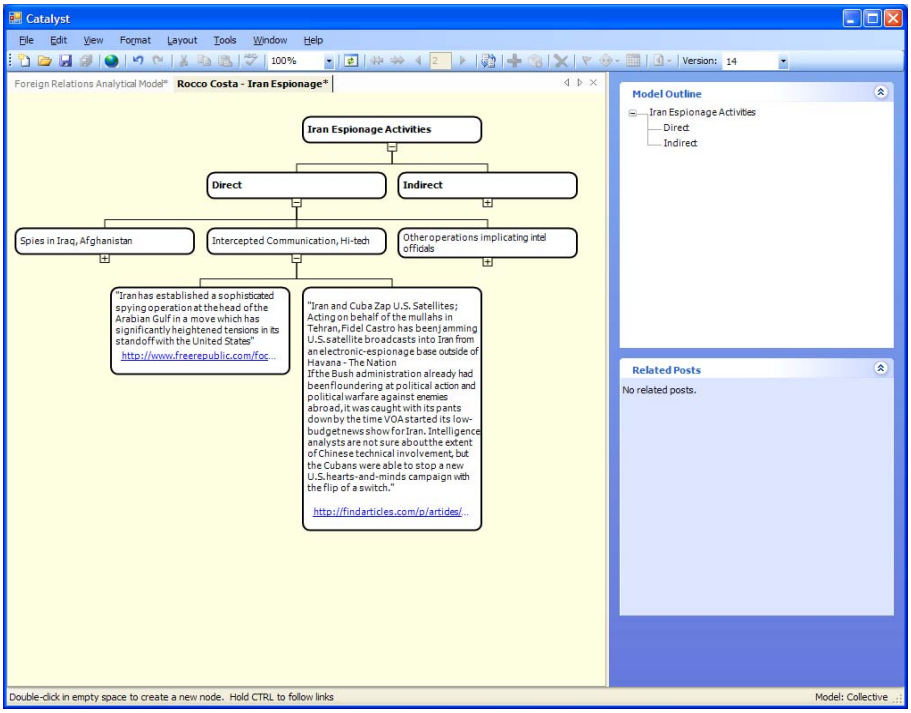

Figure 8. Catalyst – vertical layout view

- Reveal Attribution: A "view option" for Catalyst is to show (or not) attribution. When this option is selected, each node's author's name is shown below the node's text.

- Title Nodes: Nodes in a Catalyst tree can be designated as "title nodes." (Although only the root node and any other nodes that are children of title nodes can be designated as such.) When a node is designated as a title node, two things happen: 1) its text is bolded, and 2) an entry for the node appears in the Model Outline, described below.
- Model Outline: The Model Outline is a separate, small summary view that is available for Catalyst models. It appears as a small window in the upper right corner of the Catalyst application, and is intended to provide an overview of the key topic nodes in a Catalyst model—that is, the nodes that a user has designated as "title nodes" (as described above). The model outline also facilitates navigation of the model. For example, selecting a node in the model outline causes the model on the main canvas to expand to show that node, if it had been hidden.
- "Copy as Wiki Markup" Function: Catalyst provides a convenient mechanism for getting information from Catalyst into a MediaWiki-based wiki (such as Intellipedia). An analyst can right click on any node and can then select the "Copy as Wiki Markup" option, which copies that node and all its children to the clipboard. However, it copies the text in a special wiki "markup" format[4]. In particular, when the copied Catalyst nodes are pasted into a wiki's edit page and then viewed:
    - Title nodes in Catalyst become section headings in the wiki. And, furthermore, the "level" of the section headings corresponds to the indented structure of the title nodes in the Catalyst model.
    - Attachments in Catalyst automatically become proper references in the wiki (as in Intellipedia and Wikipedia). In particular, the titles of all attachments are listed below a "References" section header that appears at the end of the wiki entry, with reference numbers appearing in the body of the text (immediately following the text that came from each Catalyst node that contained an attachment).
- Full History: Catalyst maintains a full history of all model states as of each time any model was saved.[5] There are two tools for revealing these histories:
    - A control icon allows users to select earlier versions of the model as indexed by revision number (starting with 1 and incremented by one for each save). If the control icon is selected and active, users can "scroll" thought the revisions using the mouse wheel to see how the model evolved over time.
    - A "Show History" window is available from the "View" drop-down menu. When selected, it allows side-by-side comparison of any two versions of the selected node, much in the same manner is comparing versions can be done in a wiki. The General Dynamics Team would like to extend this capability to show differences across *models*, rather than just nodes.

[4] We use the markup for MediaWiki wikis.

[5] As is discussed in the following section, the Catalyst database is updated only when changes to a model are "saved" and written from the Catalyst client to the Catalyst database on the Catalyst server.

- Show User Activity for this Model: This window is available from the "View" drop-down menu. It provides a compressed timeline for each user showing activity levels via small shaded vertical bars for time increments, with darker colored bars representing greater levels of activity. It provides a quick and high-level view of who worked on what models when.
- Show Changes: This selection is available from the "View" drop-down menu. When activated, all nodes are colored based on how recently they were added or last modified, making it easy to see what is new and what has been recently changed. (Dark green indicates older nodes; brighter green, newer nodes.)
- Web View: To provide access to Catalyst models via a Web browser, we have developed a rather rudimentary read-only Web view of Catalyst. *Figure 9* shows the same Catalyst model as in *Figure 7* and *Figure 8*.

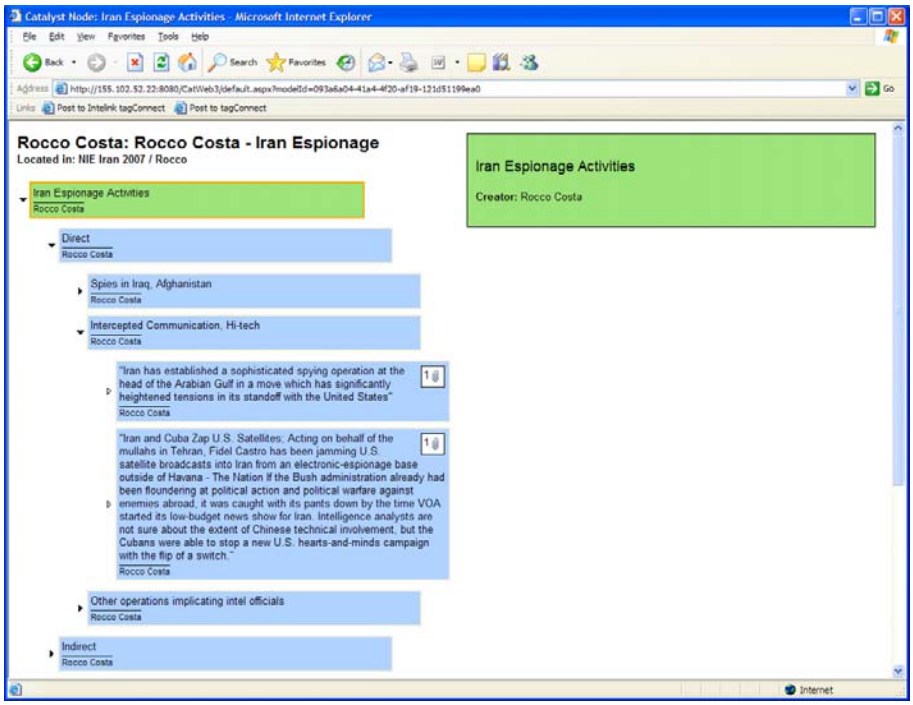

Figure 9. Catalyst Web view

2.2.3 Catalyst as a Shared Resource

As noted above, Catalyst is a client-server application: all Catalyst models are stored on a central server and are accessible by others. Furthermore, Catalyst models can be either "individual models," which are available for all to see (and to "adopt" from), but which only the creator/owner can modify, and "collective models," which anyone with access to Catalyst can add to or modify.

There are three basic issues that must be addressed for any client-server system that allows multiple individuals to simultaneously access or update the client-side data store:

1. Server-Side Updates: How and when to update the server-side data store when a user has made local (client-side) updates or changes.
2. Client-Side Updates: How and when to update a user's (client-side) view when information in the server-side data store has been changed by someone else.
3. Server-Side Conflicts: How to handle the conflicts that can potentially arise when multiple individuals make client-side changes to "the same" server-side data.

Our approach to these issues within Catalyst is as follows:

Server-Side Updates: The three basic options for server-side updates are:

- Automatically "write" users' work to the server when the user makes any change or update[6];
- Only "write" users' work to the server when users explicitly invoke the "Save" function; and
- A hybrid, with "auto-save," where saves can be performed manually, but will also be performed automatically by the system at a pre-selected frequency (such as, say, every five minutes).

We selected option 2, giving full users control over when the server-side data is updated. We felt that because all Catalyst data on the server is available to all Catalyst users, individuals would be uncomfortable with any approach that automatically made their every keystroke (potentially) available to others for viewing.

If users try to close any model (or to exit the application) with unsaved changes, they will be prompted with an "are you sure…?" message and given a chance to save any client-side changes made since their last save to the server-side Catalyst database.

Client Side Updates: Recall that an analyst may open others' individual models for viewing, which the owner may potentially update or change at any time. And individuals may be participating in collective models, which of course others may update as well. In either case, it's quite likely that these server-side updates will occur while the analyst has the model open. We decided it would be distracting for analysts to have the model they have open seem to change "on its own" if updates were automatically pushed out to their clients and the clients' UIs.

Therefore, the approach we have taken with Catalyst can be characterized as "inform, and ask." When a model that someone has open is updated by someone else (that is, when someone else saves their changes to that model to the server), everyone else who has that model open will, within a second or two, see a small message in the upper left corner of the model window indicating that:

> "A newer version of the model exists. Click here to refresh the model." (As shown in *Figure 10*.)

[6] Quicken, for example, works this way. It doesn't have a "Save" function. Instead, all updates are stored automatically, as they are made.

There is also a “refresh” icon in the toolbar that is grayed out where there are no updates to the open model pending, and which, conversely, is shown in full color when there *are* updates pending.

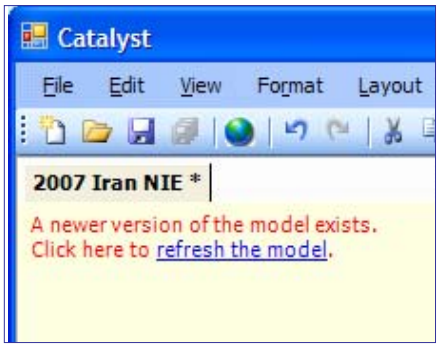

Figure 10. Catalyst’s "newer version" message

Server-Side Conflicts: Dealing with server-side conflicts is an unavoidable necessity for any client-server system that allows multiple users to access and update the same data. This is an issue for Catalyst’s “collective” models, which multiple individuals can access and update simultaneously.

There has been much research into this issue and, in general, there unfortunately is no “single best” approach. There are, however, a number of alternatives, each of which has pros and cons.

One approach is to allow write access by only *one individual at a time*. A particular variant of this approach is sometimes referred to as “baton passing,” in which only one individual can posses a single symbolic “baton” at a time, and in which that individual is the only one who can make changes to the shared space.[7] This is a popular approach for collaborative “shared whiteboards,” where only one person can have the “pen” at a time. We did not feel that this approach was at all appropriate for Catalyst, where it seems critical to let analysts get work done at their own pace, independent of what others might be doing.

Rather than avoiding conflicts by preventing simultaneous updates by multiple individuals, we decided to allow individuals to proceed at their own pace, and to have Catalyst provide assistance if and when conflicts do occur.

We should point out, though, that we have designed Catalyst to minimize any such conflicts by having the minimal possible scope for updates, by which we mean that if one user modifies a particular node (changing its text, moving its location, adding an attachment, etc.), there will be a conflict only if someone else tried to modify *that very same node*. Thus, two analysts can work without conflict on a single model as long they

[7] Any number of mechanisms can control who gets the baton next, including a queue or having the current holder of the baton decide to whom to release it.

work on different subtrees within that model.[8] This stands in contrast with other shared access collaborative systems, such as a wiki, where the level of granularity is either the entire wiki page or an individual subsection of that page (depending upon which edit mode the user selects).

However, despite Catalyst's fine-grained approach to updating, there will be times when conflicts do occur. Our approach is to have Catalyst *inform* the user of the conflict and to *guide* the user through a decision and action process to resolve or address the conflict.

An example: Let's say that "Ann" and "Bill" are working on the same collective model at the same time. Ann modifies the text in a particular node at the same time that Bill is modifying the text in that same node. Ann and Bill, however, are making different modifications reflecting their evidently differing views on how the node's text should be worded. Let's say that Ann saves her work before Bill does. Shortly after she does, Bill receives the following message in the upper left corner of his Catalyst canvas:

> "A newer version of the model exists. You should save your changes to check for conflicts. Or click here to refresh the model."

If he clicks on refresh the model, he will loose his changes. So, instead he clicks on the "Save" button. At this point the message in the upper left changes to:

> "The model failed to save. Newer versions of the conflicted nodes exist in the database. Please copy the conflicting nodes and refresh the model."

Also, the node he had been editing, with edits that are now in conflict with the version that Ann has already saved to the database, is shown in Red. Bill follows the suggestion, and copies and pastes the node to another location on the canvas. He then clicks on the refresh the model link in the message that still appears in the upper left. Catalyst pops up a message asking him it he would like to save his work, and he answers "yes."

Now, the Catalyst canvas shows Ann's update (she saved her work first, after all) *as well as the copy that Bill created of the node that was in conflict with Ann's edits.*

This approach certainly doesn't in any sense fully *resolve* the conflict, but it at least provides Bill with an opportunity prevent his work from being overwritten by Ann's edits and therefore lost. And, furthermore, Bill can now take his time to think through how he should respond to Ann's edits: Should he accept her edit in favor of his? Should he add his thoughts somewhere else in the model to make sure that his work doesn't get lost? Should he engage in a discussion with Ann to figure out how to proceed? The fact that these options are still on the table—rather than being lost if Bill's edits were to be overwritten by Ann's—is the key point.

[8] We are not suggesting that analysts need to coordinate the sections of the model on which they are working (although they certainly can if they wish to), only that our approach minimizes the likelihood of inadvertent conflicts when analysts *don't* coordinate with each other.

Although Catalyst's current support in helping analysts address editing conflicts is indeed functional and useful, we believe that with additional work and development it can be made substantially more user-friendly and "transparent."

2.2.4 Trees as a "Balance Point" Among Representations

We selected a tree structure as the fundamental organizing principle for Catalyst because we believe it provides a good balance between expressivity on one hand, and "readability" on the other. Fully expressive systems tend to be less "readable," so for collaborative systems, where people must to at least some degree access and understand each others' work, getting this balance "right" is particularly critical.

Regarding "readability": Tree structures are familiar: we are all used to creating and reading outlines and other hierarchical decompositions. And in Catalyst, in particular, analysts can capture in upper-level nodes their approach a particular issue, challenge, or problem.

In Catalyst we provide a number of UI mechanisms to allow the Catalyst tree structures to be collapsed and expanded, precisely so that individuals *can* quickly and easily get a high-level sense of what a particular Catalyst model is about, and can also drill in (by expanding selected sub-trees) to get more detail, as needed. Ideally, the high-level structure provides a "guide" on where to drill in, just as does the table of contents in a reference book. (This collapse/expand capability is valuable not only to the tree structure's creator, but also to individuals who want to see and leverage each others' work.)

Regarding expressivity: A tree structure is only one of many that can be used to organize information and thinking. Alternative structures that could be used are "general graphs" and "directed graphs" (which are general graphs that additionally specify directions—parent-child relationships—among nodes).[9]

"Concept Maps" are a particular form of a general graph that are both expressive and flexible. However, they arguably lack readability. Concept maps aren't required to have "starting points," as do trees with their root nodes, and thus it can be difficult to get "oriented" in one. Also, as a related shortcoming, they lack the collapse/expand capability of Catalyst's trees.

In summary, there is no "right or wrong" with representations, only tradeoffs. For a collaborative system, we decided to place a strong emphasis on readability, arguably at the expense of expressivity, to ensure not only that the system is easy to learn and use, but also to ensure that individuals' work *is* "readable" by others. We believe that Catalyst's tree structures *do* provide a sufficient and powerful expressivity for a wide variety of analytic challenges.

[9] Trees are "directed graphs," with the restriction that each node has no more than a single parent. In a general directed graph, no such restriction holds, and, as a result, any node can have any number of parents, potentially creating cycles (loops].

A sobering account of the challenges in getting this balance "right" is provided by Conklin, et al,[10] who developed the Compendium collaborative software application, which has roots going back the Issue-Based Information Systems (IBIS) developed in the 1970s, and which allows users to employ a small set of node types that can be connected into directed graphs via a small set of link types.

Compendium's developers attempted to develop an absolutely minimal set of formalisms in support of their design goal of providing transparent and intuitive support to the thinking and design processes. Nonetheless, they concluded:

> "A primary lesson from these early experiments is that the effort required to think and represent hypertextually is comparable to the development of fluency in a new language—it is a whole new literacy."

Ultimately, Compendium became a tool targeted for facilitators' use in face-to-face meetings. This put the load of developing this "fluency in a new language," and of employing this fluency effectively, on the facilitators, who use the tool—projected on a screen—to capture and guide the group's collaborative thinking process.

Given our objective that Catalyst *not* require a facilitator, we believe that it is absolutely critical to keep the representation as simple as possible, hence our decision to use trees rather than more general relational structures.

2.2.5 A Brief History of Catalyst

2.2.5.1 CIM

The General Dynamics Team's first ideas and inspirations for Catalyst arose while we were developing an earlier software tool called CIM, or Critical Intent Modeler. The idea of CIM was to allow an analyst to lay out, in a tree form, a set of options or potential courses of action presumed to be open to an adversary. Analysts then established various *attributes* for the selected options (such as their cost, effectiveness, etc.), and then estimated the presumed *priorities* of those attributes (from the adversary's perspective), which are grouped into "scenarios." Given all this input, CIM then calculated the most likely outcome based on the assumed options, attributes, and priorities. (Analysts could test the effect of different sets of an adversary's presumed priorities—that is, different "scenarios"—in a "what if" type of analysis.)

CIM began as a "standalone" tool, intended to support individual analysts. Later, it was embedded in Groove Network's Groove[tm] collaborative, peer-to-peer environment, which allowed individuals to share a "single" model, with anyone's updates (to their copy of the model) being automatically propagated to others' copies via Groove's peer-to-peer infrastructure. It became clear that while this capability made for good demos, it actually provided rather poor support to the collaborative analytic process. In particular, it suffered from the "(s)he who writes last, wins," problem—in which anyone can delete or

[10] Conklin, et al., *Facilitated hypertext for collective sensemaking: 15 years on from gIBIS*, Proceedings of the twelfth ACM conference on Hypertext and Hypermedia, ACM Press, 2001.

modify anyone else's work—that plagues quite a few collaborative systems. This is arguably a reasonable trait for shared whiteboards, which support a group in moving forward together, and where the "costs" of having one person's work lost as the result of someone else changing or deleting it may be considered small.[11]

Through its work with CIM, the General Dynamics Team became particularly focused on the goal of supporting collective diversity, which seemed lacking in the Groove-enabled "collaborative" CIM.

2.2.5.2 M-CIM

Our initial thoughts, in March, 2003, were to develop what we then referred to as "Uber-CIM," in which a set of independent but tightly-joined CIM models could be developed. However, although that approach indeed would have "preserved diversity," it unfortunately would have made the process of achieving a meaningful consensus difficult. In particular, although it would afford each user with his or her "own" model, there was no provision for any sort of "collective," shared model.

After working thorough these issues and considering and prototyping various alternatives, in May, 2003, we developed the "M-CIM" (for "Multi-Perspective CIM") concept, which would support, essentially, a hybrid of an "individual" and "collective" model, where, at a fine-grained level, individuals could make a choice to "agree" or "disagree" regarding the individual components of a particular model. If individuals disagreed, they were free to create alternative components that were, in the data structure, explicitly identified as alternative components. While we were developing M-CIM, we also developed a generalized approach to support multi-perspective modeling for data structures and applications other than M-CIM, which we termed "M-Modeling."[12]

The fundamental goal for M-CIM and M-Modeling was to support not only "divergence" (when people hold differing opinions on an issue), but also "convergence" (when people can identify points of agreement).

Clearly, two people who disagree on an issue could create two separate—and differing—CIM models. The idea of M-CIM, though, was to allow individuals who were collaborating on an issue to identify those points where they did *agree*, and to perhaps "merge" their viewpoints together (at least partially, if not completely). Conversely, two individuals could be building an M-CIM model together, in complete agreement and synchrony, but, if and when they did *disagree* on an issue, they could explicitly represent their differing opinions specifically on only those points over which they disagree, and could do so within the context of a single M-CIM model.

---

[11] Interestingly, wikis also operate on this principle, but in some sense they make up for it by allowing reversions to recover otherwise lost work. CIM, however, did not have any such explicit support for rolling back changes.

[12] M-Modeling is Patent Pending. Sickels, et al., *System and method for multi-perspective collaborative modeling*, U.S. Patent Application 20050222836, October 6, 2005.

The initial abstract specification for M-Modeling was developed in July, 2003. We developed a test M-Modeling application in that timeframe as well, and completed the first fully-functional prototype of M-CIM in February, 2004.

The M-CIM UI could be switched back and forth between two basic layouts: one which showed a view of the user's own work, and the other with a side-by-side view of the user's "perspective" on a model (that is, how that individual believes it should be constructed) in the left-hand sub-window and someone else's "perspective" on that same model in the right-hand sub-window. (*Any* perspective on the model could be loaded for viewing into right-hand sub-window.) *Figure 11* shows both layouts, with the single-perspective view on the left, and the multi-perspective view on the right.

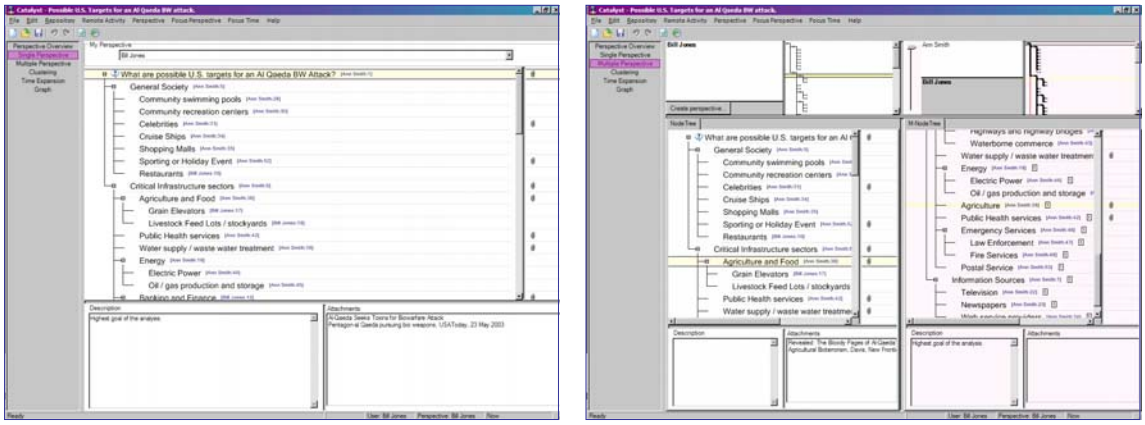

Figure 11. M-CIM: single perspective view, and multi-perspective view

The UI provided the capability for individuals to "merge" their models, selectively, thus allowing selective consensus while also preserving diversity and constructive conflict where appropriate. It was also possible (based on the underlying M-Modeling support) for individuals to "merge" items with different names, thus indicating that the two differently-named items share a common identity. This allowed individuals to indicate, for example, "you call it 'car,' I call it 'automobile,' but we're actually referring to the same thing."

Individuals could also "adopt" content from others' perspectives into their own. Not only was the adopted content added to the adopter's perspective, but in addition a linkage was created indicating that the adopted content was shared between the two perspectives.

The General Dynamics Team did extensive prototyping and experimentation with M-CIM, and including substantial work with a variety of UI configurations and features. Ultimately, however, we decided to move away from our M-Modeling-based approach to M-CIM. The following issues drove this decision:

User and Analytic Issues:

- It is actually quite challenging for individuals to "merge" two models together (selectively or otherwise)—not so much, we believe, from a UI/interaction

perspective, but actually from a *cognitive* perspective. Also, users found the results and outcome of their efforts to be less compelling than we had hoped.

- As intended, M-CIM supported a midway point, of sorts, between the extremes of a single collective "consensus" model, on one hand, and a collection of separate, individual models, on the other. However, a lingering issue was that at the end of the day (even after cross-perspective adoptions and merges), a group of collaborating individuals would still have a *collection* of individual (albeit overlapping and interlinked) "perspectives" on a particular analytic issue rather than any sort of "single view." And for better or worse, it is a "single view" (perhaps with caveats and alternatives explicitly pointed out) that a policy maker or tasker would likely want to receive in response to his or her request.

Technical Issues:

- The M-Modeling data structure represents selectively entwined perspectives and the evolution of those perspectives over time. As such, it is quite complex. This underlying complexity, in turn, made it difficult for the General Dynamics Team to update and modify the software code quickly in response to new prototyping needs and requirements.
- Finally, because of the nature and complexity of the data structure, performance of the application was unacceptably slow.[13]

For the above reasons, we decided in March, 2005, to re-implement M-CIM with a simpler underlying data structure.

Regarding the name "M-CIM": With M-CIM we had shifted away from CIM's focus on calculating most likely outcomes based on options, attributes and priorities to a new focus on representing diversity and convergence in a collaborative setting. We therefore decided in July, 2005, to rename M-CIM as Catalyst 1.0. We also named the then-forthcoming *reimplementation* of M-CIM as Catalyst 2.0.

2.2.5.3 Catalyst 2.0

As one of our first steps in developing Catalyst 2.0, in July, 2005, the General Dynamics Team conducted a "whiteboard" experiment, in which together we created a collective "model" addressing a specific question. The model was created from "nodes" that the participants hand-wrote on paper, cut out with scissors, and taped to a whiteboard in a tree-based configuration. The participants could add their own nodes, and also could take turns moving (or deleting) others' nodes. Individuals could also "vote" on nodes (by

[13] In particular, the M-Modeling data structure is represented via numerous internal "pointers" that create connections among perspectives as well as connections across time. In order for the M-CIM "engine" to retrieve any particular view of the data for rendering, it had to follow numerous chains of pointers, which led to unacceptably slow application performance. We were able to improve the application's performance somewhat, but not sufficiently, and not for all queries that engine needed to make to retrieve data for rendering.

placing red or green check marks beside them) to indicate their importance or relevance to the question at hand.

The goal of this experiment was to establish a baseline for how a group might perform the particular task of constructing a "consensus model" in a face-to-face setting, without any computer support, in order to uncover any dynamics that would likely need to be addressed in our then-forthcoming computer-based implementation. Then in October, 2005, we conducted the "same" experiment with our initial prototype implementation of Catalyst 2.0, with the participants sitting at their computers in their individual offices, but otherwise performing the "same" operations as they had in the July whiteboard experiment. *Figure 12* shows one individual's view of multiple models from the October experiment. One feature that we did include in Catalyst 2.0—to at least somewhat make up for the lack of face-to-face contact and communications that we had had in our whiteboard experiment—was an integrated chat function.

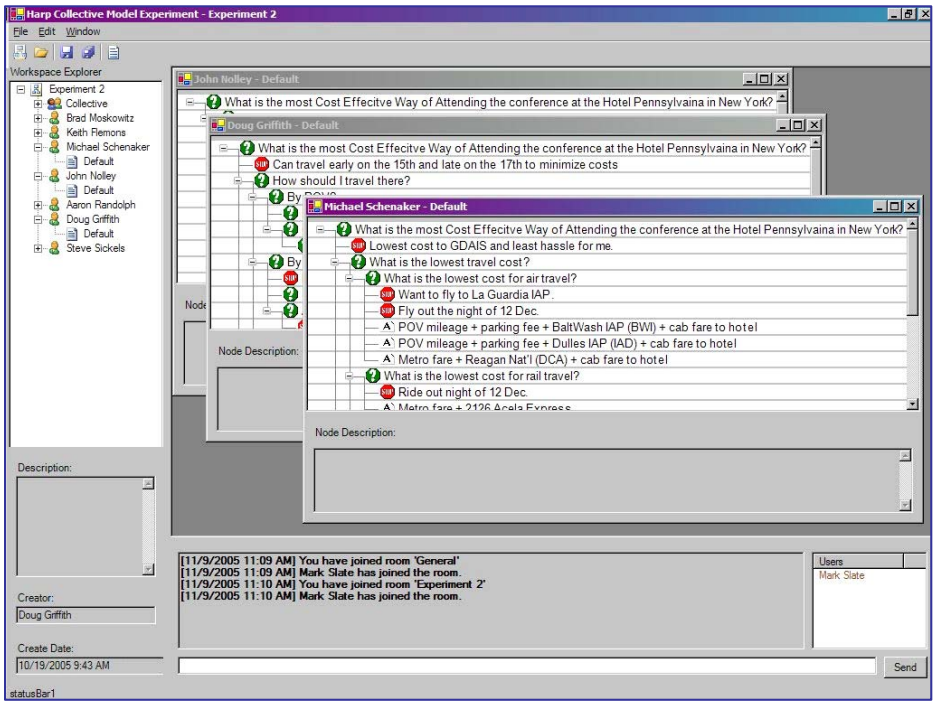

Figure 12. Catalyst 2.0, with multiple models open

Also, Catalyst 2.0 contained a "filter" such that nodes with vote scores below a threshold could be filtered out and not shown. (Green checks added 1 to a node's score, and red checks subtracted 1 from a node's score.) *Figure 13* shows a filtered consensus view from that experiment.

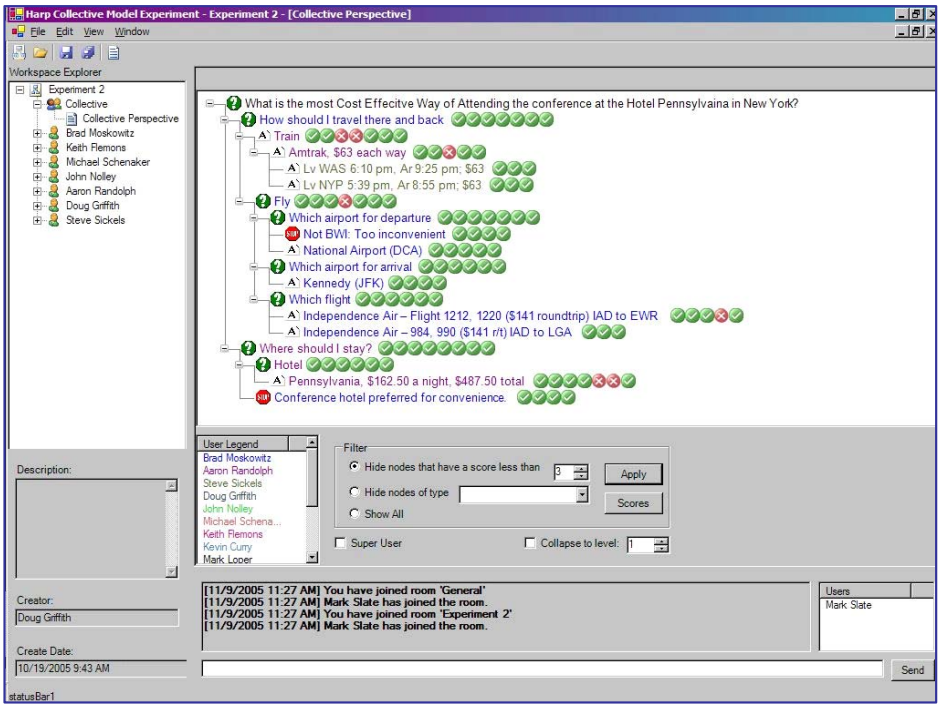

Figure 13. Catalyst 2.0, with filtered consensus model

Based on the October, 2005, experiment, Catalyst 2.0 could probably be best characterized as a "mixed success." It did indeed allow the development of a single "Consensus Model"—that is, the model filtered such that nodes with low scores were not shown. In our experiment the filtered collective model was coherent and did seem to reflect the group's consensus.

There were, however, some conceptual difficulties with the voting approach: Individuals remained a bit unsure of just what they were voting on, and evidently had difficulty in creating an effective mental link between what voting *means*, on one hand, and the *effect* that votes had on a (filtered) model, on the other. Put in other terms, the idea of "voting" on nodes to ensure that they would show up in a final model seemed at odds with people's intuitive sense of a vote as a choice of *the single best option* from among a set of alternatives. (Ultimately, for these reasons, we dropped the voting feature from Catalyst.)

The General Dynamics Team continued to work on improving Catalyst—its functionality, user-friendliness, and robustness—up through May, 2006, when we kicked off a ten-week experiment with our subcontractor the Monterey Institute for International Studies to assess the utility of both Catalyst 2.0 and tag|Connect. And, most importantly, to get feedback to fold back into our development process. (This experiment is further described in Section 3.4, below.)

We received a wealth of valuable feedback from the experiment. Overall, the results were encouraging, but the three biggest "negatives" (things to improve) were:

- Voting: User feedback and observations of the participants' work confirmed that the participants were not fully comfortable using a voting process as a mechanism for developing a "consensus" model.
- Discussions: It was clear from user feedback and observations of the participants' work that a means for "discussing" Catalyst model content was needed. Ultimately, we responded to this need by developing the Context-Grounded Conversations described in Section 2.3, below.
- Robustness: As a client-server application supporting shared, simultaneous access to both "individual" and "collective" models, Catalyst is a complex application that can be used in innumerable ways. Despite our having performed extensive testing in advance of the experiment, during the experiment Catalyst repeatedly threw exceptions and/or crashed. We pushed out updates with bug fixes frequently. However, at the end of the experiment we still did not consider Catalyst to be sufficiently robust.

We continued to add functions to Catalyst 2.0—such as an XML export/import capability—up through February, 2007, at which time we began working on a completely new, "refactored"[14] implementation, with a greatly simplified architecture, to address the robustness issues.

2.2.5.4 Catalyst 3.0

Development of Catalyst 3.0, shown in *Figure 5* through *Figure 8*, began in February, 2007. Changes from Catalyst 2.0 include:

- Unique Identifiers: Full support for GUIDs (Globally Unique Identifiers).[15]
- Workspaces: Although the "workspace" concept from Catalyst 2.0 was quite powerful (it supported *virtual* collections of models), we decided to switch to a more familiar file-based paradigm. (Catalyst models are actually stored in a database, rather than in files. But in Catalyst 3.0 they *appear* as files to users.)
- Data Access: While our intention with Catalyst 2.0 was to enforce a clean separation between the business logic and the data access logic, they were in fact still a bit mixed due to Catalyst 2.0's fine-grained data access interface. In Catalyst 3.0, the data access interface was instead at the level of models and directories, which helped simplify the code as well as allowing developers using the code to work with simpler objects.
- Models and Nodes: We simplified the data structure such that models are made up only of nodes, rather than of nodes and edges, as in Catalyst 2.0. This better supported collaborative interaction *and* revision control (as in a wiki).

[14] "Refactored" is developer-speak for restructuring and rewriting source code to improve internal consistency, performance, extensibility, etc., without changing its functions.
[15] This is a requirement for Intelink deployment.

- Revision Control: Revision control is handled in the repository. Whenever a model is saved, a new revision of that model is created.
- Undo and Redo: One of the major issues during the experiment was the model getting "out of synch" due to bugs in the Undo/Redo feature, which was implemented at the UI level. In Catalyst 3.0, undo support is built directly into the API, which greatly improves its functioning.

Finally, several (of many) changes to the UI include:

- Support for multiple trees on the canvas of a single "model."
- A "Model Overview" window for model navigation and orientation. This small window—rendered inside the main Catalyst UI window—shows the tree structure of the "title nodes" that have been so-designated by the Catalyst users.

For a more complete listing of Catalyst's features, please refer to Section 2.2.2, above.

2.2.5.5 Catalyst 3.5

As of December, 2007, we are taking the initial steps in the implementation of our next version of Catalyst, 3.5, which will be a completely new Web-based UI built on top of the "business logic" we have already developed for Catalyst 3.0. We plan to use the new "WPF" technology from Microsoft, and when version 2.0 is released in March, 2007, Microsoft's Silverlight technology, which provides for a Rich Internet Application (RIA) in a browser.

2.3 Context-Grounded Conversations

2.3.1 Overview

Context-Grounded Conversations are computer-based discussions—or "conversations"—that are explicitly linked to *content* represented within an application. The conversation and the content are shown together, even though they are managed by two separate applications (that is, the conversation application, such as a Web-based discussion forum, and the content application, such as Catalyst).

A given discussion that is linked to a particular application's content can be viewed from within *either* application:

- *Viewed from within the content application* (such as Catalyst): Any discussions linked to the application's content are viewable next to the content to which they refer.
- *Viewed from within the discussion application* (such as a Web-based discussion forum): A small "snippet" of application content is shown next to the discussion, thereby allowing readers to get a quick sense of what is being discussed without having to switch to the content application to see what is being discussed.

This explicit linkage, rendered from within both applications, provides a clear "grounding" of—as well as an explicit context for—discussions, hence the name "Context-Grounded Conversation," or "CGCs." These linkages allow the important

"social infrastructure" of queries, requests, suggestions, challenges, and negotiations to be brought to bear in the development of application content. And they allow that social infrastructure to be enhanced and informed by the structure that can be provided by shared application content.

Furthermore, because the discussion applications are "loosely coupled" to the content applications, a given discussion application—such as Web-based discussion forums, Instant Messaging (IM), or email—can be linked to *multiple content applications*. And, the content within a given content application—such as Catalyst or Intellipedia—can be discussed via *multiple discussion applications*.

Our first key goal with CGCs is to *extend* discussion mechanisms that people *already use* to support explicit linkages between application content and discussions about that content. By integrating CGCs into a larger, pre-existing fabric of computer-based communications, discussions can be conducted naturally, organized by topic or theme, and can reference application content if and when appropriate. This stands in contrast to applications in which a discussion capability is *directly embedded* within the application framework and is designed only for discussing content within that application's framework.[16] Any direct embedding is *awkward* because users have to switch from their "standard" communication mechanisms to the embedded communication mechanisms whenever they want to easily discuss specific application content.[17] And a direct embedding is *limiting* because discussions embedded directly in content application frameworks are not accessible from general discussion contexts, and vice versa.

Our second key goal is that CGCs be generally available across a range of discussion mechanism and content applications:

- *Discussion mechanisms* can include not only discussion forums, but also IMs, email, and perhaps other communication modalities.
- *Content* that can be discussed will include not only Catalyst models and nodes, but also items tagged in tag|Connect. We anticipate developing a mechanism for discussing Intellipedia content, as well as. Our intention that both we and others can develop CGC capabilities for multiple content applications.

In the subsections below, we address our discussion forum implementation for CGCs, followed by our initial work on an Instant Messaging CGC application. And finally, we discuss a number of general aspects of our implementation.

2.3.2 Implementation – Discussion Forum

General Dynamics' first implementation of Context-Grounded Conversations is a Web-based discussion forum (the discussion mechanism) and Catalyst (the content application).

[16] This is the approach taken within wikis, such as Wikipedia and Intellipedia, where discussions are conducted on a special "discussion" tab associated with each wiki page.

[17] Users can make written references in their general discussions to "such-and-such a file," or "such-and-such a wiki page, but these are time-consuming to create and awkward to follow.

- *Launching and Viewing Conversations from within Catalyst*: Catalyst users can launch discussions directly from within Catalyst by right-clicking on any node and selecting the "Discuss this…" option, which causes a special Catalyst window to pop up into which an analyst can type her comments. As soon as she hits "enter," her post is written—via the discussion forum's API—to the discussion forum database and is then immediately available to anyone viewing the forum. Currently it's possible to see, in Catalyst, all "Related Posts"—that is, a listing of all discussion forum posts that refer to the currently-open Catalyst model. Clicking on any of the listed post launches the full post in the Web-based discussion forum.
- *Viewing Conversations from within the Discussion Forum*: Any thread or post that is linked to a Catalyst model will automatically include a small snippet of the Catalyst model, rendered adjacent to the post, as shown in *Figure 14*. This snippet provides readers of the forum with a quick sense of what is being discussed, and also provides a hyperlink that launches the full Catalyst model (currently via the read-only Web view).

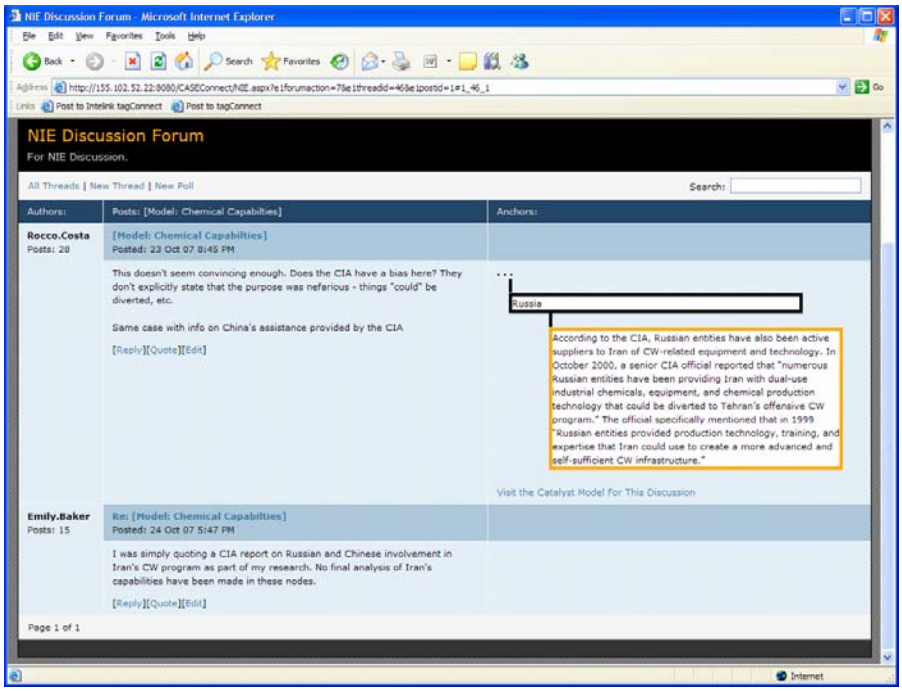

Figure 14. Context-Grounded Conversation - discussion forum implementation

- *Contributing to discussions from within the Discussion Forum*: If someone accessing the discussion forum wants to add a post that includes a link to a Catalyst model (perhaps responding to someone else's comments with a note about a relevant Catalyst model), he can get a special link from within Catalyst—again, by right-clicking on a Catalyst node, but this time selecting a "Copy Context Snippet" option, which adds the link to the user's clipboard. The user then pastes the content of his

clipboard (containing the link) into a special field in the discussion forum's "create a post" input form. Anytime that post is rendered, the snippet is shown as well.

Regarding viewing conversations from within Catalyst: As noted above, currently in Catalyst it's possible to see a listing of all threads that contain references to the open Catalyst model. (The lower right-hand Catalyst sub-window shown in *Figure 5* lists one related post for the model shown.) There are, however, a number of improvements we plan on making to dramatically increase the power of Context-Grounded Conversations within Catalyst:

- Posts that reference particular Catalyst nodes should be viewable adjacent to or somehow directly associated with the nodes to which they refer. The implementation details need to be worked out. However, we will develop some sort of a window that pops up next to a node that shows the post's text and that is clearly associated with that post.
- We plan on developing an improved capability to show users, at a glance, what nodes are being discussed. (Perhaps even what nodes have been discussed most recently.) In general, we believe that there are tremendous opportunities to provide a variety of visual queues, overlaid upon Catalyst models, to help collaborative participants see which parts of a model are being discussed and what has changed. With these sorts of additional information, Catalyst can become a true "map" of an analytic effort.

2.3.3 Implementation – Instant Messaging

As of this writing, the General Dynamics Team has created the initial building blocks for a CGC implementation for Instant Messaging. We have chosen the "Jabber" instant messaging protocol—both because it is an open standard, as well as because it is currently in use within the Intelligence Community. (Technically, Jabber is a standard based on XMPP, or eXtensible Messaging and Presence Protocol. In practice the two terms are often used interchangeably.)

We have developed an "extension" to the Jabber standard (via an extended namespace) that allows the same HTML snippets that we described above for our discussion forum implementation to be encapsulated within the XML of Jabber messages.

We have also developed an initial prototype of a modified browser-based, open source Jabber client that understands the extension and that has two added windows: one for entering the snippets, and the other for displaying them. All users with this CGC-enabled Jabber client will be able to send and receive Catalyst snippets along with their messages.

This client is shown in *Figure 15*, below.

Regarding Jabber "extensions": the Jabber protocol specifies that support for extended namespaces is *optional*, and that if a Jabber client receives a message with content conforming to an unsupported extension, the extension should be *ignored*.[18] This means, practically speaking, that CGC messages can be sent to *any* Jabber client, with the caveat

[18] See Section 2.4 of the XMPP protocol, at http://www.xmpp.org/rfcs/rfc3921.htm, for details on extensions and how they are handled.

that the snippet payload will be ignored by non-CGC-aware Jabber clients. And, of course our CGC-extended client will receive Jabber messages from any Jabber client (extended or not). Thus, we are not in any way creating any sort of *separate communications channel* for Context-Grounded Instant Messages; instead, we are, in essence, "riding on top of" a popular standards-based Instant Messaging protocol—and we are fully in conformance with that protocol, via its allowance for extensions.

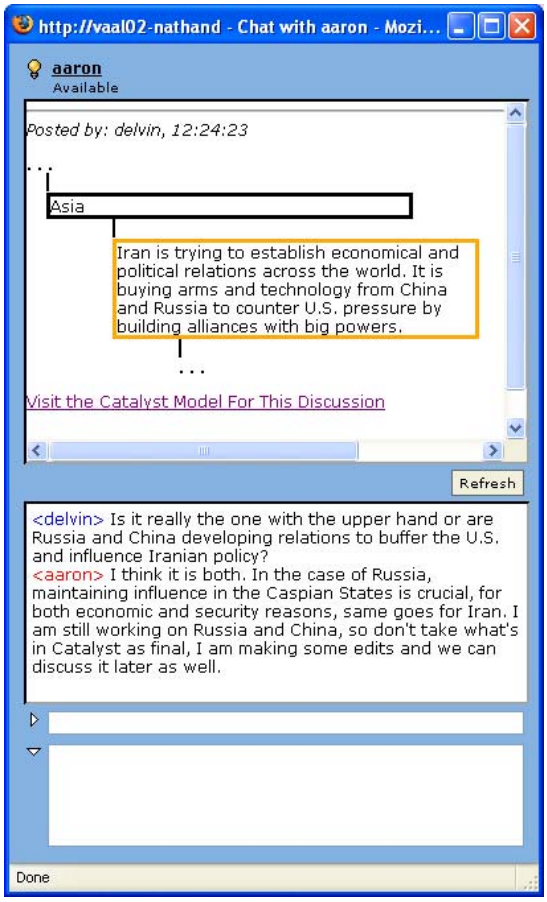

Figure 15. CGC-modified Jabber client

As noted above, we have only begun developing a CGC implementation for Instant Messaging, so there are a number of open issues that still need to be worked out. For example, Jabber supports both synchronous and asynchronous messaging, and we need to make sure that our client presents the snippets in an intuitive manner for both cases—in particular ensuring that the queue of messages is synched with the queue of snippets for the asynchronous case.

We also need to incorporate Jabber messaging (and a Jabber client) directly into Catalyst, such that Catalyst users can right-click on a node to launch an Instant Messaging session regarding that node, just as they can currently right click on a node to launch a threaded discussion about that node via our discussion forum CGC implementation.

2.3.4 Implementation – General

We noted above our intention to allow CGCs to be easily extended to new discussion mechanisms, and to allow content from applications other than Catalyst to be discussed via CGCs.

Toward that end, are developing Web Services-based standards to handle interactions and information exchange between the content and discussion applications. In our Catalyst / Discussion Forum implementation, we are currently making direct programmatic calls from within Catalyst to the discussion forum's functionality. However, by abstracting the functionality of both the content application side and the discussion application side into Web Services APIs, it should be much easier, in the future, to extend other content applications *and* discussion applications to support Context-Grounded Conversations.

2.3.4.1 "Discussion-Side" Services

The following "discussion-side" services will be specified to provide clean interfaces with which "content-side" applications (such as Catalyst) can access already-developed discussion-side CGC functionality:

- *New Thread Service*: When a Catalyst user right-clicks on a node and selects "Discuss this…," Catalyst will accesses the discussion forum's "New Thread" service and will pass to it the user's input.[19] This same service will be available for other content applications to access as well.
- *New IM Service*: When a Catalyst user right-clicks on a node and selects "IM this…," Catalyst will access the IM's "New IM" service and will pass to it the user's input. Again, as with the "New Thread" service, this service will be available for other content applications as well.

2.3.4.2 "Content-Side" Services

The following "content-side" service will be specified (and implemented by Catalyst):

Snippet Service: For Catalyst, we have designed an HTML snippet that depicts the Catalyst node that's being referenced (and its parent node for a bit of additional context), as shown in *Figure 14* and *Figure 15*. The Catalyst application is responsible for generating the snippet, and the discussion forum application is responsible for displaying it.

Currently, when a user selects "Discuss this…" for a node from within Catalyst, Catalyst makes an internal call (via .NET remoting) to the discussion forum, passing it the snippet HTML. To implement this exchange via a new Web Services-based snippet service, we would change things slightly: In particular, after Catalyst calls the New Thread (or New IM) service, the New Thread (or New IM) mechanism on the discussion side would in turn call back to Catalyst, requesting the snippet via the Web Services-based Snippet API.

By making this change and developing this API, it will be far easier for a new content application to support CGCs. The requirement will be that the content application must return HTML—that in some way provides some context for users—in response to any calls to their Snippet API. It would then be completely up to the content application developer to determine what that HTML should look like.

To fully decouple the snippet provider (the content application) from the snippet consumer (the discussion mechanism), we anticipate developing a Web Services-based standard by which the provider and the consumer can negotiate at least some parameters of the snippet. This would allow, for example, a larger or more comprehensive snippet to

[19] The snippet communication mechanism is not yet fully specified. It is likely that "content" applications that provide snippets will be expected to provide them on demand, in response to Web Services calls. Such as when someone is viewing a discussion application that contains a content application link.

be shown if both the provider and consumer can support it. If either can't, though, the preview should fall back to the maximal capability of the "least capable" of the two.[20]

2.3.4.3 "Content-Side" Application Modifications

We believe that the future success of CGCs is strongly dependent upon the ability of "content-side" applications (such as Catalyst) to make CGCs appear, from the user's perspective, to be well-integrated into the content-side application. Getting this right requires work. And, for Catalyst, we still have more to do. In particular, as noted above, the General Dynamics Team needs to develop a capability within the Catalyst UI for users to see any individual post in a way such that it is clearly associated with the node to which it refers. Or, put another way, users should be able to easily see—if they choose to—any discussion that is "about" particular content element, such that it is quite clear what discussions *are* about that content.

If this visual coupling of content and discussion can be accomplished, then CGCs can achieve a "best of both worlds" success:

On one hand, CGCs can make it easy for individuals to engage in "embedded" discussions of specific application content, as well as making it easy for individuals to access and see what others are discussing (or have discussed), without the need for shifting back and forth between two applications.

On the other hand, by being built on "standard" communications mechanisms, CGCs can be mixed in, as needed, with more general communications. Conversations can then flow easily, with specific application contexts invoked only when needed. There are collective benefits as well (at least for the discussion forum, which is a collective discussion mechanism): anyone viewing the forum can read and see posts with snippets, and can follow the links from the snippets to the full application contexts. This might be done even by individuals who are not aware of the content application and/or particular content within that application. This then can really open up the value of the applications and application content to individuals who would otherwise have been completely unaware of them.

3 "Understanding" Through Experimentation

At the heart of our "Develop / Understand / Improve" cycle, introduced in Section 1, above, are the processes by which we "understand" the collaborative, analytic, and technical foundations of the tools we develop. The knowledge and understanding we gain through these processes drives our development process, and is, we believe, critical and necessary to our success.

Intelligence analysis experts and practitioners have participated directly and integrally in our efforts:

---

[20] This will be a much-simplified version of the approach outlined within the Open Geospatial Consortium's "Web Service Common Implementation Specification" (http://www.opengeospatial.org/standards/common) , which allows geospatial content providers and consumers to "automatically" negotiate the form of the geospatial data to be exchanged.

- Experiments: With our subcontractor the Monterey Institute for International Studies, Monterey Terrorism Research and Education Program (MIIS MonTREP), we have performed a number of experiments with our tools that have provided us with invaluable insights and understanding, which have, in turn, informed and driven our development process.
- Advisors: A number of advisors with deep expertise in intelligence analysis have played key advisory roles in work. In addition to their participation in the design and conduct of the above-mentioned experiments, these advisors have played key roles in the formation and development of the conceptual foundations upon which our development efforts are built.

3.1 Experiments - Overview

We have conducted a number of successful "experiments" with our subcontractor MIIS MonTREP:

- 2007: Beginning in October, 2007, we conducted a nine-week experiment with ten MIIS MonTREP graduate students. The tools employed were Catalyst, tag|Connect, a discussion forum that we augmented to support CGCs, and a MediaWiki wiki.
- 2006: Beginning in May, 2006, we conducted a ten-week experiment with seventeen MIIS MonTREP graduate students and staff, and with four Department of Homeland Security interns. The tools employed were Catalyst, tag|Connect, and an open-source discussion forum.
- 2005: Beginning in September, 2005, we conducted a seven-week experiment with twenty MIIS MonTREP graduate students and researchers that was specifically focused on tag|Connect.

Each of these experiments is described in more detail below.

Although in each experiment there has been a clear focus on the General Dynamics Teams' tools' usability, we have also been particularly interested in developing an understanding of *processes* by which the tools are employed and how the tools' use does—or doesn't—support the end-goal of effective collaborative analysis.

In particular, because a central goal of our work is providing effective support to *collaborative* analysis, we have been particularly interested in how the participants deal with a host of tradeoffs that any collaborative group must face, including individual vs. collective work, and diversity vs. consensus. Observing how the participants have—and haven't—balanced these sometimes-competing needs, as well as the individual and collective strategies they employed in accomplishing their tasks, has been enormously informative for our development team.

So the sorts of questions we're interested in addressing through our experiments include not only rather concrete questions about our tools, such as how to improve the intuitiveness of Catalyst's user interface, but also more abstract issues that have more to do with collaborative *processes* than with any specific details of our tools'

implementations. For example, in our 2007 experiment we were particularly interested in the extent to which individuals can make use of each others' individual Catalyst models in the performance of a collaborative task.

Getting insights into these sorts of analytic and collaborative process issues and the tradeoffs associated with them has been instrumental in the development and refinement of our conceptual foundations:

- Blending the personal and the collective: The capability of a tool to provide collective benefits from analysts' "individual" work.
- Capturing diversity: The capability of a tool to capture disagreement, dissent, and differing views.
- Context-grounded consensus formation: The capability of a tool to support a process by which a "meaningful" (and not forced) consensus is developed.

Following each experiment we have analyzed results and outcomes (which include not only the participants' work, but also questionnaire responses and follow-up interviews). We have then identified strengths and shortcomings along collaborative, analytic, and technical dimensions. And from those strengths and shortcomings, we developed prioritized lists of "next steps" for further development, which then drove our forthcoming work (and forthcoming experimentation).

3.2 Experimental Design

The goal of all of our experiments has been to learn as much as possible along three rather broad dimensions (collaborative, analytic, and technical) to inform and drive our software development process. Our experiments have therefore all been somewhat exploratory in nature. This stands in some contrast to a more rigorous experimental framework that would be predicated on clearly-defined hypotheses to be tested and a carefully-bounded experimental agenda.

And, furthermore, because of the central importance of the *collaborative* dimension to our research, we have always opted to maximize the number of participants who were using our tools, rather than dividing the available pool of participants into an experimental group and a control group.

Finally, both tag|Connect and Catalyst support analysts in building up information over time that can then be leveraged in a collaborative analytic process. We have therefore chosen to conduct one two-month experiment per year, where we can test how information organized in our two tools supports further collaborative research and analysis.

3.3 2007 Experiment

Beginning in October, 2007, we conducted a nine-week experiment with ten MIIS MonTREP graduate students. The tools employed were Catalyst 3.0, tag|Connect, a discussion forum that we augmented to support CGCs, and a MediaWiki wiki.

Although it was our intention to learn as much as possible about all the tools, our primary interest was in Catalyst 3.0.

The tasking given to the participants was: "What are the most important and most likely threats to the security of the United States and its interests abroad that could arise from the Islamic Republic of Iran within the next 10 years?"

The participants were to use all the tools, and were to produce "a cogent, fully-sourced and cited, summary document reflecting their analysis of this issue" in the form of a National Intelligence Estimate (NIE), which was to be developed in the wiki, and which was to be no more that 10,000 words in length. A full set of Terms and References was provided to define terms and establish various parameters for the experiment.

The participants were allowed to develop their own schedule (to meet the specified due date) and to organize their collaborative work as they saw fit. Gary Ackerman and other advisors from CETIS were "on call" for any help the participants might request, as well as to monitor the proceedings and to intervene if necessary.

However, during training, a high-level workflow was suggested: begin by researching sources and tagging them in tag|Connect. Then shift to developing "descriptive" models in Catalyst.[21] Then shift to developing "analytic" models in Catalyst. Finally, use Catalyst's "Copy to Wiki" function to export the Catalyst models to the wiki, and continue work to produce the required final product. Given that high-level workflow, however, the participants had tremendous freedom to decide, for example, how and when to produce individual vs. collective models, and when to transition from Catalyst to the wiki.

3.3.1 Lessons Learned

Through the participants' work (all of which was captured on the General Dynamics server that hosted the four applications), and through their survey results and follow-up interviews, we gained tremendous insights into the application of our tools to a collaborative analytic process in general, and many specific technical ideas as well.

An extensive analytical report on the experiment has been developed by Gary Ackerman, James Fobes, and Charles Blair, all of CETIS.

A very selective and abbreviated listing of a few key observations and conclusions is as follows:

Catalyst 3.0: The participants appreciated Catalyst's power as a collaborative tool. However, they felt that Catalyst models became hard to read when the models grew large or when the nodes themselves contained extensive text. As a result, the participants for the most part moved directly from developing individual models to doing their collective

[21] During pre-experiment training, we discussed two possible "types" of Catalyst modes: "descriptive" and "analytical." We were clear during the training that these were not hard distinctions, and that any such distinctions were in no way prescriptive of how Catalyst should be used. The terms were intended only to provide an indication of a range of uses to which Catalyst could be applied.

and collaborative work in the wiki, thereby "skipping" the step of developing collaborative analytic models in Catalyst. In general, the participants felt that their work would have been better had they been able to and/or chosen to do more work in Catalyst, vs. in the wiki.

As a result of this experiment and feedback, we have a number of ideas that we would like to prototype which we believe will vastly improve Catalyst's readability.

Context-Grounded Conversations (CGCs): Not all users tried this feature out, but several did. There was a consensus, though, that the CGCs that did occur improved the quality of the analysis, and that CGCs are a valuable analytic tool. There was strong agreement that CGCs need to be integrated more fully (from the UI perspective) into Catalyst.

Organization: Because of the rather broad nature of the participants' tasking, the participants chose individual focus areas that did not have much overlap. As a result, there was less chance for the students to collaborate on specific issues. The participants were consistent in their view that this hurt the quality of their analysis.

Production: As noted above, the participants switched to the wiki rather early in the overall process. And they had tremendous problems with editing conflicts in the wiki (an issue that Catalyst is specifically designed to minimize). They therefore selected three "writers" who were responsible for editing the wiki. This caused problems, though, because the wiki had grown to well above the 10,000-word limit, and quite a bit of cutting was needed. The cuts were not coordinated, though, which left some participants feeling that their contributions were misrepresented or left out.

Estimative Language: Related to the above issues were some difficulties with the NIE's estimative language. Some of the participants had proposed language that distinguished between uncertainty due to a lack of evidence vs. uncertainly due to a lack of consensus. However, the language that they ended up adopting—which was modeled after that in a declassified NIE—did not make this distinction clear. This conflation tended to wash out the presentation of disagreements in the group, which arguably lessens the value of the NIE to a policy maker or decision maker.

Conclusions: We believe that many of the issues encountered will be addressed by Catalyst when its ability to support the viewing and understanding of large models and large nodes is improved. In particular, we believe that Catalyst will be particularly effective at supporting the preservation of diversity within a collaborative task environment. And that improved Context Grounded Conversations—including the Instant Message-based CGCs that we have begun to develop—will provide excellent support in letting users work through the challenges of paring work down through collective participation.

3.4 2006 Experiment

Beginning in May, 2006, we conducted a ten-week experiment with seventeen MIIS MonTREP graduate students and staff, and with four Department of Homeland Security

interns. The tools employed were Catalyst 2.0, tag|Connect, and an "out of the box" open-source discussion forum.

The goals for the experiment were to see if and how the improved tag|Connect supported intelligence analysis, and to test, for the first time with MIIS MonTREP, Catalyst as an analytic tool. Additionally, we were interested in determining the analysts' ability to self-organize. This last objective was focused on highlighting analytic and collaborative process requirements, at the possible expense of analytic quality.

The tasking given to the participants was: "In light of a possible U.S. (or U.S. led) confrontation with Iran over the alleged Iranian nuclear weapons program, consider the potential behavior of Hizb'allah and the security implications thereof."

The participants were to develop and vote on a final consensus model, using Catalyst 2.0's voting and filtering mechanisms. Upon completing the final consensus model, the participants were to develop a final analytic report.

The final 31-page report was titled *Hizb'allah's Response to a U.S. or U.S.-led action Against Iran*. The General Dynamics Team prepared a separate report on the experiment and our conclusions from the experiment.

3.4.1 Lessons Learned

As with the 2007 experiment, through the participants' work, and through their survey results, we gained tremendous insights into the application of our tools to a collaborative analytic process in general, and many specific technical ideas as well. A selective and abbreviated listing of a few key observations and conclusions is as follows:

Catalyst 2.0: The participants appreciated Catalyst's power. Although there was not a full consensus on Catalyst, the participant's responses to the survey questions indicated that they did indeed see its power to support collaborative analysis.

Several quotes from the questionnaires:

> "I liked the ability to see the ideas, thoughts, assumptions, etc. of other project members on the given subject."
>
> "I liked the fact that [Catalyst] was able to give you a visual representation of an analytical project, rather than to have to see the same type of analysis portrayed through pages and pages of research. It highlights key points and questions that both you and group members feel strongly about and it facilitates the process of conceptualizing."
>
> "The best thing about Catalyst is that it allows people to be able to sort out their thoughts on a computer in a way that makes it easy to find and sort through information. Although Catalyst cannot substitute for a face-to-face meeting, it is nice to see other people's viewpoints in one place. Also, it is very convenient to be able to view attachments and evidence for specific points, with commentary on the attachments all in the same place. It saves a lot of time to look at someone's comments and go directly to the relevant place in the article."

However, despite these positive statements, it became quite clear that Catalyst 2.0 was not robust enough to provide effective support to collaborative analysis. This realization led directly to the refactoring of Catalyst that we undertook following the experiment.

Catalyst 2.0 Voting: The participants tried using the Catalyst 2.0's voting system to create a filtered-down consensus model. In general, the participants had a difficult time applying a "voting" process to achieve a filtered model, when the general connotation of "voting" is to choose the single best option from among a set of alternatives. This difficulty was compounded by our decision to have votes "propagate" up the tree, which we did to ensure that parent nodes would not be filtered out.[22] Ultimately, we took the decision that our voting mechanism, even if implemented "properly"—that is, without propagation—was confusing and that it didn't achieve the objective for which we designed it. We therefore took it out.

Conversations: At the time of the experiment we were already quite interested in developing what we later named "Context-Grounded Conversations." However, we did not have time prior to the experiment to implement any such feature. What we did, instead, was to provide: a "chat" feature in Catalyst; a "private message" feature (like email) in tag|Connect; and also a separate discussion forum capability. We knew that having three separate communication functions was far from ideal, but we wanted, nonetheless, to provide at least some discussion capability.

Despite Catalyst having a chat feature, there are quite a number of nodes in the Catalyst model that were, effectively, *comments* on other nodes. This had the unfortunate effect of "cluttering up" the model and making it much harder for the participants to easily see the model's core analytic content. Seeing this cluttering certainly strengthened our conviction that some sort of integrated—and, ideally, context-grounded—communications capability would be a valuable addition to our tools. This conclusion was also consistently reinforced by the survey results. Several quotes:

> "No efficient way of communicating with team members."
>
> "… an important feature that is missing for quick and effective communication is another program that will overlook tag|Connect, Catalyst, …"
>
> "… an integrated chat feature between tag|Connect and Catalyst… [would] exponentially increase the tools' collaborative capacity…"
>
> "… easy to use messaging system integrated into both tools…"

tag|Connect: In general, tag|Connect was well-liked and well-used by the participants. They did provide us with quite a few suggestions that we later implemented which improved its utility significantly.

[22] We realized that this was an issue partway through the experiment, and removed the propagation, function and adjusted the filtering algorithm such that all parents of a high-scoring child node would not be filtered out. However, unfortunately, at that point the model already had been "polluted" with votes automatically applied via propagation, vs. by individuals' choice. The fact that the model then contained different "types" of votes further muddied their meaning.

Conclusions: In general the participants endorsed our approach. Despite software difficulties and the need to self-organize, the participants did work collectively and did benefit from their collective work. However, the need for self-organization hindered collaboration. Furthermore there was confusion over voting and how to build models.

The requirements that came out of the experiment were:

- Catalyst needs to be more robust,
- The "voting" approach did not prove to be effective,
- An integrated communications capability is key, and
- More process guidance for the participants would have helped.

3.5 2005 Experiment

Beginning in September, 2005, we conducted a seven-week experiment with twenty MIIS MonTREP graduate students and researchers that was specifically focused on tag|Connect (which then went by a different name).

There were two key goals for this experiment, both of which were met:

- Determine how tagging can work in the context of an analytic tasking "out of the box." (That is, a fairly close approximation to the "del.icio.us" social bookmarking / tagging system on the Web.[23])
- Develop specific ideas for improving the current approach and finding broader uses for tagging within the Intelligence Community.

The tasking given to the participants was: "Investigate the possibility that Islamist terrorist groups might collude on an operational level with elements of radical anti-U.S., anti-Semitic, or anti-capitalist groups, whatever their specific ideology. Research, analyze, and compose an assessment of this threat."

The participants were to deliver this assessment in the form of a document.

3.5.1 Lessons Learned

The students felt that tag|Connect provided both individual and collective benefits. It was this experience that effectively established the "blending of the personal and the collective" theme that has continued to inform and guide all of our work since.

In general the participants picked tag|Connect up quite quickly. Some participants, however, started off creating tags by concatenating terms representing multiple concepts together, rather than breaking them up into separate tags. After we explained that their ability to drill in and explore the "tag space" would be far greater if they avoided concatenating terms, they (mostly) switched to using separate tags for separate concepts.

[23] http://del.icio.us

This experience emphasized that at least a bit of training (more than we provided) would likely be beneficial to the group.

We identified an area that was (and is) not well supported in del.icio.us, which is clearly of key importance to the Intelligence Community: Finding other people who have a particular interest in a particular tag or tags. We worked to address this requirement through our subsequent work on the "graph view" of the tag|Connect. Ultimately, though, a number of factors prevented this work from being realized. These factors included awkwardness in the available tools designed for displaying and manipulating graphs[24], and issues of scalability—that is, how to provide consistently meaningful and understandable displays when there may be only a few people with related interests, or there may be thousands. In the most recent versions of tag|Connect we have—we believe—begun to address this need with a "Related Users" tab (an alternative to the "Related Tags" tab) that is available on the tag|Connect page for any tag or combination of tags.

3.5.1.1 Publications

The following paper, prepared as part of this effort, describes the 2005 experiment and the lessons learned in more detail:

> Ackerman, Gary; James, Molly; and Getz, Casey. "The Application of Social Bookmarking to the National Intelligence Domain." *International Journal of Intelligence and Counterintelligence* 20, no. 4 (Winter 2007): 678-698.

3.6 Advisors

The Advisors on the General Dynamics Team have played a key role in our processes—and in our successes. All advisors have been involved directly in the development, specification, and conduct of our experiments. And have provided advice and guidance in the nature and challenges of collaborative analysis.

Gary Ackerman: Mr. Ackerman is currently the Director of the Center for Terrorism and Intelligence Studies (CETIS), and was formerly the Deputy Director of MIIS MonTREP. Following Mr. Ackerman's departure from MIIS MonTREP, General Dynamics established a subcontract with CETIS, who in turn established a subcontract with MIIS. Mr. Ackerman has deep experience in terrorism studies and risk analysis, and has had primary responsibility for developing and executing all of our experiments. He has, on numerous occasions, contributed idea and suggestions regarding our tools and how they can be improved.

Steven Simon: Mr. Simon is the award-winning coauthor of *The Age of Sacred Terror* and *The Next Attack*. Former director for global issues and senior director for transnational threats at the National Security Council (NSC). He is currently a Senior Fellow for Middle Eastern Studies at the Council for Foreign Relations. Mr. Simon

[24] Visualization toolkits we worked with included the open source and Java-based *Jung* and the commercial and .NET-based *goDiagram*. Both toolkits have substantial drawbacks and limitations.

participated on our effort during the 2004 and 2005 timeframe, via a subcontract that we held with Cyberneutics, Inc. He was able to provide invaluable insights into the role of intelligence analysis in supporting the policy maker, which was his role on the NSC.

3.6.1.1 Publications

The following paper, co-authored by Mr. Simon with Mr. Hart of Cyberneutics, and prepared under this effort, describes a range of issues that negatively impact the intelligence analysis process:

> Hart, Douglas; and Simon, Steven. "Thinking Straight and Talking Straight: Problems of Intelligence Analysis," *Survival* 48, 2006:1, 35-59.

Among their many suggestions for improving intelligence analysis are a number of technological suggestions that we have pursued, directly, on this effort. A quote:

> Critical thinking can be enabled by collaboration, especially when it involves compiling, evaluating and combining multi-disciplinary perspectives on complex problems. Effective collaboration, however, is possible only when analysts can generate and evaluate alternative and competing positions, views, hypotheses and ideas.
>
> Enforced consensus relegating alternative assessments to footnotes – for example, in documents like the National Intelligence Estimates – has been a disincentive to collaboration within the intelligence community's analytical corps. In addition, collaboration and sharing generally require extra work that competes with time spent on individual assignments. Interim and final analytical products have to be recast in a format that is accessible to all members of a collaborative team as well as compatible with their individual toolsets. What is needed are tools that encourage the blending of the personal and the collective through shared access by allowing analysts to easily shift focus within a common context from individual work to work being done by others, and by making the work of others directly available for 'adoption'. As a result, analysts, especially junior analysts, would be able to generate and evaluate alternative and competing ideas, avoid groupthink, and reach consensus while making use of all available analytic assets.

4 Glossary

AFRL	Air Force Research Laboratory
AJAX	Asynchronous JavaScript and XML: Web technology that improves Web pages' interactivity
API	Application Programming Interface: The specification by which a software component accepts requests from and responds to requests from other software components
Calls	An informal term for internal requests made by one software component to another component's API. The requesting component is often an application's UI, which generally makes requests of the application's engine, but requests can originate from other applications as well.
CASE	Collaboration and Analyst System Effectiveness: An IARPA R&D program
CAST	An application developed by Lockheed Martin, focused on user modeling
Catalyst	General Dynamics-developed collaborative tool, developed on this effort
CETIS	Center for Terrorism and Intelligence Studies: Subcontractor to General Dynamics on this effort.
CGC	Context Grounded Conversation: A General Dynamics-developed approach to computer-based discussion capabilities that are explicitly linked to analytic artifacts, such as Catalyst models.
CIM	Critical Intent Modeler: An analytic tool developed by General Dynamics that is particularly focused on evaluating an adversary's presumed options
DAL	Data Access Layer: The components of a software application that are responsible for mediating communications between the engine and the data store
Engine	An informal term for those components of a software application that, collectively, process requests. These components should generally be distinct from the UI and the Data Access Layer (DAL) components.
HTTP	Hypertext Transfer Protocol: A communications protocol used to transfer information on the Web
IARPA	Intelligence Advanced Research Projects Activity: The ODNI's R&D office
ICES	Intelligence Community Enterprise Services: The office within the ODNI CIO's office that manages the Intelink Networks
IMO	Intelink Management Office: the former name for what is now ICES

Intelink	The umbrella term for the three IC-wide internets: JWICS, SIPRNet, and Intelink-U, all of which are hosted and operated by ICES
Intelink-U	The unclassifed Intelink network
Intellipedia	The wiki implementation that can be found on the Intelink networks
ITE	Intelink Technical Exchange: Technical Exchange Meeting hosted by ICES
Jabber	An open standard and specification for Instant Messaging. Closely related to XMPP, the protocol upon which it is based
JWICS	The Top Secret-level Intelink network
LDAP	Lightweight Directory Access Protocol: A common and popular application protocol for querying and modifying information about a computer network's users and network resources
MIIS	Monterey Institute for International Studies: Subcontractor to General Dynamics on this effort
MonTREP	Monterey Terrorism Research and Education Program: The MIIS Program with whom we worked on this effort
NSC	National Security Council
ODNI	Office of the Director of National Intelligence
OSW	Open Source Works: CIA-operated open source analysis center
REST	Representational State Transfer: A software architectural-approach for transferring Web Services data (and other domain-specific data) over HTTP
RDEC	Research and Development Experimental Collaboration: ODNI Program for deploying and testing tools for the Intelligence Community
RIA	Rich Internet Application: A general term for a relatively new class of browser-based applications that support a high-level of interactivity
RSS	Really Simple Syndication: a Web feed format, supporting subscriptions to Web-based content
Silverlight	Microsoft technology that supports Rich Internet Applications (RIAs)
SIPRNet	The Secret-level Intelink network
Tag space	An informal term connoting a collection of tagged items and the tags applied to those items
tag|Connect	General Dynamics-developed collaborative tool, developed on this effort
UI	User Interface

URL	Universal Resource Locator (more properly, Universal Resource Indicator, or URI): The “address” that allows an item to be retrieved by a Web browser (this is an imprecise definition, but conveys the central idea of URLs)
WPF	Windows Presentation Format: A Microsoft technology that supports a new approach to developing application UIs. Silverlight is a subset of WPF intended specifically for the development of browser-based applications
XML	Extensible Markup Language: A general purpose markup language whose primary use is sharing structured data via the Internet
XMPP	Extensible Messaging and Presence Protocol: A standard protocol and specification for Instant Messaging. Closely related to Jabber, an Instant Messaging standard that is built on XMPP